AD-A058 413 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO F/G 12/1
AN ADAPTIVE FINITE-DIFFERENCE METHOD FOR SOLVING THE STURM-LIOU--ETC(U)
MAY 78 D A NELSON
UNCLASSIFIED AFIT-CI-78-22 NL

1 OF 2
ADA
058413

78-72 D

(14) AFIT-CI-78-22 LEVEL II (1)

The Pennsylvania State University

The Graduate School

Department of Computer Science

(6) An Adaptive Finite-Difference Method for Solving the Sturm-Liouville Problem.

A Thesis in

Computer Science

by

(10) David Alan/Nelson

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

(9) Doctoral thesis,

(12) 191p.

Doctor of Philosophy

(11) May 1978

012200

78 08 31 009

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER CI 78-72	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) An Adaptive Finite-Difference Method of Solving the Sturm-Liouville Problem		5. TYPE OF REPORT & PERIOD COVERED Dissertation 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Captain David A. Nelson		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT Student at the Pennsylvania State University, University Park, PA		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS AFIT/CI WPAFB OH 45433		12. REPORT DATE 1978 13. NUMBER OF PAGES 178 Pages
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for Public Release; Distribution Unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES [illegible] JOSEPH P. HIPPS, Major, USAF Director of Information, AFIT AUG 15 1978		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)		

78 08 31 009

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

The signatories below indicate that they have read and approved the thesis of David Alan Nelson.

<u>Date of Signature</u>:	<u>Signatories</u>:
Feb 16, 1978	Charlotte F. Fischer Charlotte F. Fischer Professor of Computer Science Chairman of Committee Thesis Advisor
Feb 16, 1978	[illegible] Patrick C. Fischer Head of the Department of Computer Science
Feb. 16, 1978	Don E. Heller Don E. Heller Assistant Professor of Computer Science, Member of Doctoral Committee
Feb 16, 1978	[illegible] Wendell H. Mills, Jr. Assistant Professor of Mathematics, Member of Doctoral Committee
Feb. 16, 1978	[illegible] Jeffrey R. Spirn Assistant Professor of Computer Science, Member of Doctoral Committee

ABSTRACT

The Sturm-Liouville (S-L) boundary value problem arises frequently in the study of mathematical physics. Most problems cannot be solved analytically and therefore must be solved with one of the many numerical techniques available. The majority of the current methods use either a uniform mesh or a predetermined non-uniform mesh on which to approximate the problem. It is widely accepted that such problems can be solved more accurately and efficiently with an adaptive method.

In this thesis ~~we discuss~~ primarily the finite-difference methods are discussed for solving the S-L problem. ~~We develop~~ An adaptive finite-difference method is developed in which the final mesh of grid points depends on the solution. The non-uniform discretization and the resulting matrix problem are explored in detail by establishing all the important properties, selecting a solution method, deriving convergence results and error estimates, and examining mesh refinement criteria.

Special topics relating to the method are also treated. These include developing an automatic technique for computing an initial estimate to the solution, monitoring the solution during the iteration process to insure convergence to the correct solution, and solution of singular S-L problems.

We evaluate the performance of our algorithm in terms of space and time requirements versus the eigenvalue index k, to ensure that the method has practical application. We find that space and time do not increase prohibitively as k increases. Finally, our algorithm is compared to a current initial value method. We illustrate the greater accuracy and speed of our method.

Numerical solutions are presented for several problems, both regular and singular, along with other numerical results used to substantiate our theoretical results.

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENTS

The author expresses his gratitude to Dr. Charlotte F. Fischer, his thesis advisor, for education, motivation, and guidance during his work at The Pennsylvania State University.

Support for the author's graduate work has been provided by the United States Air Force through the Air Force Institute of Technology.

Chapter 1

INTRODUCTION

The Sturm-Liouville boundary value problem arises frequently in the study of mathematical physics. For example the problem

$$y'' + \lambda y = 0 \qquad (1.1.1)$$
$$y(0) = y(1) = 0$$

occurs in the study of the vibrating elastic string. The values of λ for which a solution exists are found to be related to the natural frequencies of the string. The problem (1.1.1) is an example of a regular Sturm-Liouville problem.

A singular problem such as

$$y'' + \{\lambda - q(x)\}y = 0$$
$$y(0) = 0,\ y(x) \text{ bounded as } x \to \infty$$

occurs in quantum mechanics. Examples include the study of nuclear shell theory and the vibrational theory of diatomic molecules. Regular and singular problems are formally defined in Chapter 2.

Many methods have been proposed to solve the Sturm-Liouville problem numerically. Among these is the general class of finite-difference or direct methods. This thesis reviews the various methods, placing particular emphasis on the finite-difference type. An automatic, adaptive finite-difference method is developed in which the mesh of grid points used in the discretization depends on the solution.

In Chapter 2 we provide the reader with necessary background material by presenting many of the classical results of Sturm-Liouville theory.

In Chapter 3 we review other methods for solving the Sturm-Liouville problem numerically. The theoretical basis of each method is presented along with discussions of recent work in each area.

The review continues into Chapter 4 with Sections 4.1 and 4.2 exploring in detail the uniform mesh finite-difference method. In Section 4.3 we develop a non-uniform mesh finite-difference method. Properties of the resulting matrix problem are established, a solution algorithm is developed, and convergence results as well as an error estimate are derived.

Chapter 5 deals with automatic solution methods. Both a non-adaptive and an adaptive algorithm are presented along with numerical results from each.

For a numerical algorithm to be completely automatic the user can be expected to supply only the problem to be solved. He must not be required to provide an estimate to the solution. To this end, in Chapter 6 we present an algorithm for computing an estimate to the solution for use as a starting value in the algorithms of Chapter 5.

In Chapter 7 we develop some additional techniques necessary to solve singular problems adaptively. Two general types of singular Sturm-Liouville problems are solved and numerical results presented. For each problem solved, the adaptive method is compared to a uniform mesh finite-difference method. We also note a further use of the adaptive method showing how it can be used to select the best transformation function for use with some uniform mesh methods.

Finally, in Chapter 8 we evaluate the performance of the method by showing the relationship between the index of the desired eigenvalue and the space and time required to compute that eigenvalue. In addition we compare its performance to an existing software package developed at Sandia Laboratories, Albuquerque, New Mexico[1].

Chapter 2

STURM-LIOUVILLE THEORY

2.1 Introduction

As was stated in the previous chapter the purpose of this thesis is to study adaptive methods for solving numerically a specific type of Sturm-Liouville system.

To solve any problem numerically it is first necessary to understand the mathematical theory of the problem. To this end we present in this chapter an overview of the theory of Sturm-Liouville systems, both regular and singular.

2.2 Regular Problems

A Sturm-Liouville (S-L) equation is a second-order, homogeneous, linear differential equation of the form

$$\frac{d}{dx}\left[p(x)\frac{dy}{dx}\right] + [\lambda r(x) - q(x)]y = 0, \quad a \le x \le b \qquad (2.2.1)$$

where λ is a parameter, p, r, and q are real-valued functions of x, and p and r are positive. We further require that p, p', r, and q be continuous on $[a,b]$.

A Sturm-Liouville system is a Sturm-Liouville equation together with boundary conditions.

A regular S-L system is an S-L equation defined on a finite interval, $[a,b]$, with boundary conditions of the form

$$A_1 y(a) + A_2 y'(a) = 0 \tag{2.2.2}$$

$$B_1 y(b) + B_2 y'(b) = 0 \tag{2.2.3}$$

where A_1, A_2, B_1, and B_2 are real constants and

$$|A_1| + |A_2| \neq 0$$

$$|B_1| + |B_2| \neq 0$$

The values of λ for which (2.2.1) has a non-trivial solution satisfying (2.2.2) and (2.2.3) are called <u>eigenvalues</u>. Any solution corresponding to an eigenvalue is called an <u>eigenfunction</u>. The set of all eigenvalues is called the <u>spectrum</u> of the system.

We note here that any equation of the form (2.2.1) can be transformed via the Liouville transformation [see Birkhoff and Rota[3], p. 104, for example] into Liouville normal form

$$\frac{d^2w}{dx^2} + [\lambda - \bar{q}(x)]w = 0$$

which is defined on the new interval [c,d]. We therefore choose to work primarily with S-L equations of the form

$$y'' + [\lambda - q(x)]y = 0. \tag{2.2.4}$$

In addition we choose to consider only boundary conditions of the first kind, i.e. conditions (2.2.2) and (2.2.3) with $A_2 = B_2 = 0$. This serves to simplify the matrix eigenvalue problems we must solve; however, extensions to other boundary conditions are possible. Fox[18] discusses some of these techniques as they relate to two-point boundary value problems.

We are now able to state the following results about S-L systems. These results are presented here as theorems, however all proofs are omitted. The reader is referred to Hille[24] or Coddington and Levinson[9] for the proofs.

The first few theorems establish the existence of solutions and some basic properties of these solutions.

THEOREM 2.1: The regular S-L system (2.2.1), (2.2.2), (2.2.3) has an infinite number of real eigenvalues $\lambda_0, \lambda_1, \lambda_2, \ldots$ forming a monotone increasing sequence with $\lambda_n \to \infty$ as $n \to \infty$.

THEOREM 2.2: The eigenfunction $y_n(x)$ corresponding to λ_n has exactly n zeros in the interval $[a,b]$ and is uniquely determined up to a constant factor.

THEOREM 2.3: Eigenfunctions of a regular S-L system corresponding to different eigenvalues are orthogonal with weight function r, i.e.

$$\int_a^b r(x)y_m(x)y_n(x)dx = 0 \quad \text{if } \lambda_m \neq \lambda_n.$$

(Note: In equation (2.2.4) $r(x) = 1$.)

The next several results show how the eigenfunctions corresponding to different eigenvalues differ in terms of zeros and maxima-minima.

<u>THEOREM 2.4</u>: Consider

$$u''(x) + F(x)u(x) = 0$$

where $F(x)$ is positive and continuous in (a,b). Let $u_1(x)$ and $u_2(x)$ be 2 linearly independent solutions of (2.2.4) and suppose that $u_1(x)$ has at least 2 zeros in (a,b). If x_1 and x_2 are 2 consecutive zeros of $u_1(x)$ then $u_2(x)$ has exactly one zero in (x_1,x_2). (i.e. zeros are interlaced)

<u>THEOREM 2.5</u>: Consider

(a) $\quad u''(x) + F(x)u(x) = 0$

(b) $\quad v''(x) + G(x)v(x) = 0$

where $F(x)$ and $G(x)$ are positive, continuous and $G(x) \geq F(x)$ in (a,b). Suppose that (a) has a solution with 2 consecutive zeros at $x = x_1$ and $x = x_2$ and that $a < x_1 < x_2 < b$. Then if $v(x)$ is a solution of (b) with a zero at $x = x_1$, $v(x)$ has at least one zero, x_3, with $x_1 < x_3 < x_2$. (i.e. the distance between consecutive zeros decreases as λ increases)

<u>THEOREM 2.6</u>: With the same assumptions on $F(x)$ and $G(x)$, let $u(x)$ and $v(x)$ be solutions of (a) and (b) such that $u(x_1) = v(x_1) = 0$, $u'(x_1) = v'(x_1) > 0$. If $u(x)$ is increasing in $[x_1,x_2)$ and reaches a maximum at

$x = x_2$ then $v(x)$ reaches a maximum at some x_3 with $x_1 < x_3 < x_2$.

The following theorem gives us a means for bounding the ith eigenvalue.

THEOREM 2.7: Consider the system (2.2.1),(2.2.2),(2.2.3) and let

$$q = \min\{q(x) : x \in [a,b]\}$$
$$Q = \max\{q(x) : x \in [a,b]\}$$

Let the system with $q(x)$ replaced by q have eigenvalues $\{\mu_i\}$ and with $q(x)$ replaced by Q, $\{\nu_i\}$. Then

$$\mu_i \le \lambda_i \le \nu_i \quad \text{for all } i.$$

2.3 Singular Problems

A singular S-L system is an S-L equation defined either on an infinite interval or on a finite interval which has a singular point of the function q at an endpoint.

This definition applies to the normal form (2.2.4). When the general equation (2.2.1) is considered, a singular system may also have a singularity or zero of p or r. The results which follow are applicable to the normal form.

We consider the case where the interval is $(0,\infty)$. As noted by Titchmarsh[40] the case of a finite interval with a singularity at one end is quite similar.

THEOREM 2.8: For every non-real value of λ the system has at least one non-trivial solution in $L^2(0,\infty)$. (i.e. a solution $y(x)$ such that $\int_0^\infty y^2(x)dx < \infty$)

This type of S-L system can be placed into one of two classes. If all solutions are in $L^2(0,\infty)$ we have the limit-circle case at ∞. If only one solution is in $L^2(0,\infty)$ we have the limit-point case at ∞.

THEOREM 2.9: If, for a particular value of λ, the equation (2.2.4) has 2 independent solutions in $L^2(0,\infty)$ then this property holds for all λ.

An important sufficient condition for the limit-point case is as follows.

THEOREM 2.10: Suppose $q(x)$ is bounded from below. Then the limit-point case holds at ∞ and the solution $y(x)$ satisfies

i) $\lim\limits_{x\to\infty} y(x)\, y'(x) = 0$

ii) $y'(x) \in L^2(0,\infty)$

iii) $|q(x)|^{\frac{1}{2}} y(x) \in L^2(0,\infty)$

As with regular problems we can say something about the eigenfunctions.

THEOREM 2.11: Square-integrable (i.e. $\in L^2(0,\infty)$) eigenfunctions u and v belonging to different eigenvalues of a singular S-L system are orthogonal with weight r whenever

$$\lim_{\substack{\alpha\to a\\ \beta\to b}} \left[u(x)v'(x) - v(x)u'(x) \right]_{x=\alpha}^{\beta} = 0.$$

The set of values of λ for which the limit-point case holds is called the <u>spectrum</u> of the S-L system.

<u>THEOREM 2.12</u>: The spectrum is a closed, infinite, and unbounded set.

The spectrum can either be a <u>pure point spectrum</u> if all points are isolated, a <u>continuous spectrum</u>, or a combination of the two.

In a regular S-L system we found that we always have a pure point spectrum, but in a singular system the nature of the spectrum depends on the function q.

<u>THEOREM 2.13</u>: If $q(x) \to \infty$ with x then there is a pure point spectrum with at most one spectral value to the left of $q_0 = \inf q(x)$.

<u>THEOREM 2.14</u>: If $q(x) \in L(0,\infty)$ then there is a continuous spectrum covering the whole positive real axis. There may be a point spectrum on the negative real axis which may have $x = 0$ as a cluster point.

<u>THEOREM 2.15</u>: Let q(x) be negative and twice continuously differentiable. Let $q'(x) < 0$, $q(x) \to \infty$, $q'(x) = 0\,[|q(x)|]^c$ for $0 < c < 3/2$ and let $q''(x)$ be ultimately of one sign. Then if $|q(x)|^{-\frac{1}{2}} \in L(0,\infty)$,

the continuous spectrum covers $(-\infty,0)$ and there may be a point spectrum on $(0,\infty)$; otherwise, if $|q(x)|^{-\frac{1}{2}} \notin L(0,\infty)$, the continuous spectrum covers the whole real axis.

Analysis of the spectrum of a singular problem is usually done on an individual basis for each function $q(x)$.

In Chapter 7 we discuss singular problems further and give examples of each type.

Chapter 3

EXISTING METHODS

3.1 Introduction

The task of solving the Sturm-Liouville problem has been the subject of much research. There have been many numerical methods proposed and these methods are vastly diverse in technique and theory. We can, however, categorize the methods into the following 6 general areas.

a. Initial value methods

b. Function space approximation

c. Conversion to an integral

d. Separation of variables

e. Approximation of coefficients

f. Finite-difference methods

This thesis is directed toward the final area, that of finite-difference methods. However, in order to provide the proper background for our work we present here a brief description of areas a.-e. In Chapter 4 we present, in greater detail than is demanded here, a description of the finite-difference methods.

3.2 Initial Value Methods

Initial value or shooting methods have received considerable attention from researchers, especially in the applied areas where specific eigenvalue problems are being solved. This is probably due to the fact that initial value theory, on which these methods rely heavily, is widely understood and good code for solving initial value problems is in abundance.

The solution of the S-L system (2.2.1) and (2.2.2) is related to an initial value problem in the following way.

For a fixed value of λ we consider

$$Ly = \lambda r(x)y \text{ , } a \le x \le b \tag{3.2.1}$$

$$A_1y(a) + A_2p(a)y'(a) = 0 \tag{3.2.2}$$

$$C_1y(a) + C_2p(a)y'(a) = 0$$

where $Ly = -(p(x)y')' + q(x)y$ and where C_1 and C_2 are arbitrarily chosen constants such that $(A_2C_1 - C_2A_1) \neq 0$.

We are assured that (3.2.1), (3.2.2) has a unique non-trivial solution. If this solution is denoted by $y(\lambda;b)$ then we can define the following function.

$$F(\lambda) = B_1y(\lambda;b) + B_2p(b)y'(\lambda;b)$$

Requiring that $F(\lambda) = 0$ is thus equivalent to forcing our solution to satisfy the boundary condition (2.2.3). So we see that $F(\lambda_n) = 0$ if and only if λ_n is an eigenvalue of (3.2.1) and $y(\lambda_n;x)$ is the corresponding eigenfunction. The S-L problem has thus been reduced to finding the roots of $F(\lambda) = 0$ and the corresponding solutions of (3.2.1). Keller[25] advocates using a fixed point iteration scheme for finding the roots. Dranoff[15] solves a mass and heat transfer problem using a shooting method and Newton's method. A more slowly converging method may be to use the bisection method.

There are many examples in the literature of each such method as well as others. Every method, however, must solve the initial value problem (3.2.1), (3.2.2) to perform the root finding procedure. The designer of a shooting method algorithm may choose any of the numerous initial value methods to solve the problem. Thus we see that initial value methods for solving the Sturm-Liouville problem are varied in technique and are dependent both on the problem to be solved and the designer of the method.

3.3 Function Space Approximation Methods

The class of methods receiving the most attention by current researchers is that of function space approximations. The procedure is to approximate the solution of the eigenvalue problem by a linear combination of linearly independent functions in some appropriately chosen function space. The coefficients are determined by requiring that some measure of the error be minimized.

For all methods described here we let $\{u_i(x)\}_{i=1}^{\infty}$ be the set of basis functions for our space of admissible functions, H.

An approximation to the solution of the eigenvalue problem can be written as

$$y_N(x) = \sum_{i=0}^{N} c_i u_i(x) \tag{3.3.1}$$

The methods to be discussed here differ in essentially two ways. First, they differ in the way in which the $\{c_i\}$ are determined and second in the choice of the space H. We describe two of the more common ways to choose the $\{c_i\}$. The possible choices of H are evident from the cited examples.

i) Rayleigh-Ritz (Galerkin)

Rayleigh noted that the eigenvalues and eigenfunctions of (2.2.1) are stationary values and critical points for

$$R[u] = N[u] / D[u]$$

where $N[u] = \int_a^b (pu'^2 + qu^2)dx$

and $D[u] = \int_a^b (ru^2)dx$

The Rayleigh-Ritz method finds the critical points and values of $D[u]$ on some finite-dimensional subspace of H, where H is the space of all functions u such that $D[u] < \infty$.

This procedure results in a matrix problem

$$A\bar{c} = \Lambda B\bar{c} \qquad (3.3.2)$$

where

$$(a_{ij}) = \int_a^b (pu_i'u_j' + qu_iu_j)dx$$

$$(b_{ij}) = \int_a^b ru_iu_jdx$$

The extrema of the Rayleigh quotient $R[u]$ are at the $y_N(x) = \Sigma c_i u_i$ whose coefficient vectors $\bar{c}$ are the eigenvectors of (3.3.2).

Birkhoff, deBoor, Swartz, and Wendroff[4] use both the 2n-dimensional cubic Hermite subspace and the (n+1)-dimensional cubic spline subspace. They found that both methods approximate eigenvalues with error $O(h^6)$ and eigenfunctions with $O(h^3)$, where $h = \max_i(x_i - x_{i-1})$ with $\{x_i\}$ being the partition on which the subspace is defined.

Earlier Courant[11] suggested using piecewise linear functions as the subspace. Birkhoff and Fix[5] give an example using the trigonometric polynomials where $u_1 = 1$, $u_{2n} = \cos(nx)$, and $u_{2n+1} = \sin(nx)$.

In each case the matrix entries must be computed by evaluating integrals and thus, while the method can be quite accurate, it can also be time consuming.

ii) Collocation

In these methods the approximate solution (3.3.1) is required to satisfy the differential equation at N distinct points including the boundary conditions, i.e.

$$Ly_N(x_i) = \lambda r(x_i)y_N(x_i), \quad i=2,3,\ldots,N-1$$
$$y_N(x_1) = y_N(x_N) = 0$$

The resulting matrix eigenvalue problem to solve for the coefficients $\{c_i\}$ is

$$(A - \lambda B)\bar{c} = 0$$

where $(a_{ij}) = Lu_j(x_i)$

$(b_{ij}) = r(x_i)u_j(x_i)$

Russell and Varah[38] discuss subspaces of (2n-1) degree B-splines and (2n-1) degree Hermites. Russell and Shampine[37] show some results using general piecewise polynomials.

3.4 Conversion to an Integral

Another technique for the S-L problem is to convert the differential equation into an integral equation. This involves finding the Green's function, $g(x,\nu)$, for the system (2.2.1), (2.2.2). The function $g(x,\nu)$ is such that the solution of (2.2.1) is given by

$$y(x) = \lambda\int_a^b g(x,\nu)r(\nu)y(\nu)d\nu$$

or

$$y(x) = \lambda\int_a^b G(x,\nu)y(\nu)d\nu. \qquad (3.4.1)$$

If a mesh $\{\nu_j\}$, $j=1,2,\ldots,J$, is now introduced on $[a,b]$, with $a \le \nu_1 < \nu_2 < \ldots < \nu_J \le b$, we can define a quadrature rule

$$Q_J\{f(\nu)\} = \sum_{i=1}^{J} \alpha_i f(\nu_i)$$

to approximate

$$Q\{f(\nu)\} = \int_a^b f(\nu)d\nu.$$

If this quadrature rule is used to approximate $y(x_i)$ by y_i at each point $x_i = \nu_i$, we get the following matrix eigenvalue problem.

$$y_i = \Lambda \sum_{j=1}^{J} \alpha_i G(x_i \nu_j) y_j, \quad i=1,2,\ldots,J$$

or $\Lambda^{-1}\bar{y} = A\bar{y}$

where $\bar{y} = (y_1, y_2, \ldots y_J)^t$

and $A_{ij} = \alpha_i G(x_i, \nu_j)$.

The problem is now solved as a matrix eigenvalue problem where the matrix A is non-symmetric and is, in general, a full matrix. Keller shows that for every λ of the integral equation (3.4.1) there exists an eigenvalue Λ^{-1} of the matrix A, correct to within an accuracy proportional to the error in the quadrature formula used.

Many excellent adaptive quadrature rules for integration are available (see Davis and Rabinowitz[12] for examples). If such a rule were used to approximate $Q\{f(\nu)\}$, the resulting S-L method could be considered adaptive.

Use of this method entails computing J integrals and J^2 constants, and the solving of a matrix problem in which A is a full matrix. For most problems this technique would be highly time consuming because the work is proportional to J^2. It is for this reason that we look to finite-difference methods for an adaptive algorithm. By selecting the proper discretization and the best solution methods, the amount of work can be made proportional to J.

3.5 Separation of Variables

This method is based on the Prufer transformation which transforms the original S-L equation (2.2.1) into two first-order equations, one of which is independent of the solution of the other. Two new variables, $t(x)$ and $\theta(x)$, are introduced as follows:

$$y(x) = t(x)\sin\theta(x)$$
$$p(x)y'(x) = t(x)\cos\theta(x)$$

Under this transformation we get, from the original problem,

$$d\theta(x)/dx = \cos^2\theta(x)/p(x) + [q(x) + \lambda r(x)]\sin^2\theta(x) \tag{3.5.1}$$

subject to the boundary conditions

$$\theta(a) = A \tag{3.5.2}$$
$$\theta(b) = B + (n-1)\pi \tag{3.5.3}$$
$$0 \le A < \pi,\ 0 < B \le \pi,\ n \text{ a non-negative integer.}$$

and

$$dt(x)/dx = t(x)\sin\theta(x)\cos\theta(x)\left[\frac{1}{p(x)} - q(x) - \lambda r(x)\right] \tag{3.5.4}$$

If $\theta(x;\lambda)$ denotes the solution of (3.5.1), satisfying (3.5.2), then the eigenvalues of the original problem are those values of λ for which

$$\theta(b;\lambda) = B + (n-1)\pi.$$

This means that the calculation of the nth eigenvalue, λ_{n-1}, is equivalent to finding the solution of

$$B + (n-1)\pi - \theta(b;\lambda) = 0. \tag{3.5.5}$$

If we define $X(x,\lambda) = \partial\theta(x;\lambda)/\partial\lambda$, then differentiating (3.5.1) we get

$$\frac{dX(x,\lambda)}{dx} = X(x,\lambda)\left[q(x) + \lambda r(x) - \frac{1}{p(x)}\right]\sin 2\theta(x;\lambda) + r(x)\sin^2\theta(x;\lambda) \tag{3.5.6}$$

$$X(a,\lambda) = 0 \tag{3.5.7}$$

and we can solve (3.5.5) with the Newton-Raphson method

$$\lambda_n^{(k+1)} = \lambda_n^{(k)} + [B + (n-1)\pi - \theta(b,\lambda_n^{(k)})]X^{-1}(b,\lambda_n^{(k)}) \tag{3.5.8}$$

The numerical algorithm for finding the (n+1)st eigenvalue then proceeds as follows:

1. Let $\lambda_n^{(0)}$ be an approximation to λ_n

Repeat the following steps for $k=0,1,\ldots$

2. Solve (3.5.1)-(3.5.2) and (3.5.6)-(3.5.7) by some reliable ODE solver.

3. Compute $\lambda_n^{(k+1)}$ by (3.5.8)

4. If $|\lambda_n^{(k+1)} - \lambda_n^{(k)}| < \epsilon$ STOP

Banks and Kurowski[2] recommend that the integration be carried out as follows:

1. Integrate (3.5.1)-(3.5.2) from a to $(a+b)/2$ to get $\theta_a(x,\lambda)$.

2. Integrate (3.5.1)-(3.5.2) from b to $(a+b)/2$ to get $\theta_b(x,\lambda)$.

We now want

$$E(\lambda_n) = \theta_a(\frac{a+b}{2},\lambda_n) - \theta_b(\frac{a+b}{2},\lambda_n) = 0$$

and a similar Newton-Raphson technique is used to solve this equation.

This is the technique that Bailey, Gordon, and Shampine[1] use in their recent paper. The method can also be considered adaptive because a variable step integrator can be used to solve for θ_a and θ_b.

Since the eigenvalues are unaffected by the Prufer transformation, the accuracy of these methods depends entirely on the accuracies of the integration and root-finding methods used.

3.6 Approximation of Coefficients

Canosa and Gomes de Oliveira[8] first used this technique to solve the one-dimensional Schrodinger equation

$$d^2y/dx - V(x)y + Ey = 0$$
$$y(0) = y(L) = 0$$

The method is based on the idea that if the potential, V(x), is replaced by a step function, a greatly simplified problem results and can be solved exactly.

V(x) is approximated as follows:

$$V(x) \approx \begin{cases} V_1 & \text{when } 0 < x < x_1 \\ V_2 & \text{when } x_1 < x < x_2 \\ \vdots & \\ V_n & \text{when } x_{n-1} < x < x_n = L \end{cases}$$

Now each interval (x_{i-1},x_i), $i=1,2,\ldots,n$ yields a problem of the form

$$d^2y/dx^2 + (E - V_i)y = 0.$$

It is well known that such an equation has a basis of solutions consisting of piecewise sine and cosine, or exponential functions. By enforcing the boundary conditions and continuity of y and y', the problem reduces to that of solving a homogeneous linear system.

The proponents of this method claim an important advantage over many other methods. Because the basis functions depend transcendentally on λ, the system which results has an infinite number of roots. In most methods for approximating eigenvalues it is only practical to approximate a small number of values with any given mesh size. For higher eigenvalues the relative error becomes prohibitively large. In the approximation of coefficients methods the relative error is much more uniform as a function of the eigenvalue index, k. This enables Preuss[35], for example, to solve one problem for λ_{100} with his method to a relative accuracy of 6.0×10^{-4}. Such an undertaking would be impractical for most other methods.

Preuss extends the above work in two directions. He considers the general S-L problem and he generalizes Canosa and Gomes de Oliveira's work to include m<u>th</u> degree polynomial approximations of the coefficient functions. Instead of solving the resulting simplified problem exactly he uses an initial value method based on Taylor's series expansions, thus eliminating the need for knowing the basis functions exactly.

Canosa and Gomes de Oliveira found their method to be O(h) for both eigenvalues and eigenfunctions, where $h = \max_i (x_{i+1} - x_i)$. Preuss shows that for mth degree approximations the order is $O(h^{m+1})$.

The final category of solution methods is finite-difference methods. In Chapter 4 we present these methods and introduce a non-uniform method which we use to solve S-L problems adaptively.

Chapter 4

FINITE-DIFFERENCE METHODS

In this chapter we describe the finite-difference method in detail. First, in Section 4.1 we introduce the simplest uniform mesh method, Stormer's method, and analyze it completely. Then, in Section 4.2, we examine a higher order method and discuss the resulting changes in convergence theorems, solution techniques, and error estimates. In Section 4.3 we present a non-uniform mesh finite-difference method which will become the foundation for our adaptive S-L solver. Results pertaining to all the important properties of the method are presented.

4.1 The Stormer Uniform Mesh Method

First we recall the problem to be solved.

$$y'' + (\lambda - q(x))y = 0 \qquad (4.1.1)$$

$$y(a) = y(b) = 0 \qquad (4.1.2)$$

where $q(x)$ is continuous in $[a,b]$, and a and b are finite.

Let the interval $[a,b]$ be subdivided as follows:

$$x_j = a + jh , \qquad j=0,1,\dots,N$$
$$h = (b-a)/(N+1)$$

We call the points $\{x_j\}_{j=0}^{N+1}$ the <u>mesh</u>.

An approximation to the solution of (4.1.1),(4.1.2) can now be obtained by replacing these expressions with a set of finite-difference equations. The first such method, Stormer's method (see Henrici[23], Chapter 7), results in finite-difference equations of the form

$$- Y_{i-1} + 2Y_i - Y_{i+1} + h^2(q(x_i) - \Lambda)Y_i = 0 \qquad (4.1.3)$$
$$i=1,2,\dots,N$$
$$Y_0 = Y_{N+1} = 0.$$

The discretized function, $\{Y_i\}$, is an approximation to some eigenfunction of (4.1.1), and Λ, an approximation to the corresponding eigenvalue. (see Keller[25], Chapter 5, for a slightly more general discussion)

If we define Y to be the N-dimensional column vector

$$Y = (Y_1, Y_2, \dots, Y_N)^t$$

and A to be the N x N matrix $A = (a_{ij})$ where

$$a_{j,j-1} = a_{j,j+1} = -1$$
$$a_{jj} = 2 + h^2 q(x_j)$$
$$a_{ij} = 0 \quad \text{if } |i-j| > 1$$

then (4.1.3) can be written as the following matrix problem.

$$AY - h^2\Lambda Y = 0 \qquad (4.1.4)$$

The matrix A has the useful properties of being both symmetric and tridiagonal.

4.1.1 Truncation Error

The local truncation error of the method (4.1.3) at x_i is defined as the amount by which the exact solution, (y,λ), fails to satisfy the discretization, i.e.

$$\begin{aligned}\tau(x_i) &= -y(x_{i-1}) + 2y(x_i) - y(x_{i+1}) + h^2\{q(x) - \lambda\}y(x_i) \\ &= -y(x_{i-1}) + 2y(x_i) - y(x_{i+1}) + h^2 y''(x_i)\end{aligned} \qquad (4.1.5)$$

Expanding $y(x_{i-1})$ and $y(x_{i+1})$ in powers of h we find that

$$\tau(x_i) = -\frac{1}{12}h^4 y^{(iv)}(\theta), \qquad x_{i-1} < \theta < x_{i+1}$$
$$i=1,2,\ldots,N$$

If we now define $\tau_i(y) = \frac{1}{h^2}\tau(x_i) = \frac{1}{12}h^2 y^{(iv)}(\theta)$ and $\bar{\tau}(\bar{y}) = (\tau_1(y), \tau_2(y), \ldots, \tau_N(y))^t$, we can write (4.1.5) as the following matrix equation.

$$A\bar{y} - h^2\lambda\bar{y} = h^2\ \bar{\tau}(\bar{y}) \qquad (4.1.6)$$
$$y(x_0) = y(x_{N+1}) = 0$$

where $\bar{y} = (y(x_1), y(x_2), \ldots, y(x_N))^t$.

4.1.2 Convergence

The matrix eigenvalue problem, (4.1.4), possesses exactly N eigenvalues while the differential equation, (4.1.1), has an infinite number. The problem then is determining which of the exact (differential equation) eigenvalues are being approximated, and how accurate that approximation is.

Keller provides us with a theorem which clarifies this point. We present it here without proof. The vector norm used is the 2-norm.

THEOREM 4.1: (Keller) For each fixed eigenvalue λ of (4.1.1) with corresponding eigenfunction $y(x)$, there exists an eigenvalue $h^2\Lambda$ of A such that

$$|\Lambda - \lambda| \le \frac{\|\bar{\tau}(\bar{y})\|}{\|\bar{y}\|}$$

It is an easy matter now (see for example Keller p. 134-135) to show that

$$|\Lambda - \lambda| \le O(h^2)$$

Thus we see that any exact eigenvalue can be approximated to a desired accuracy by taking h small enough.

4.1.3 Solution of the Matrix Problem

If one wants to compute all the eigenvalues and eigenvectors of a matrix, QR methods have become the preferred technique. Stewart[39] gives a thorough treatment of both the explicitly shifted and the implicitly shifted algorithms.

It is shown, however, in Chapter 5 that during the adaptive solution process the final matrix problem to be solved depends on the particular eigenvalue under consideration. In general, the final mesh and thus the final matrix problem will be different for each eigenvalue.

An algorithm such as QR is therefore much too expensive to be used for our purposes since we solve for only one eigenvalue from any particular matrix problem.

Instead we use a technique called Rayleigh quotient iteration. An exhaustive analysis of this technique appears in a series of papers by Ostrowski[31]. We refer to results from these papers frequently throughout this chapter.

The algorithm we use is as follows.

ALGORITHM 4.2: Let A be a real, symmetric matrix and let Λ_0 be an approximation to Λ. (Discussion of how to get such an approximation is deferred until Chapter 6.) Let ϵ be the desired tolerance level.

```
1. Let $Y_0 = (1, 1, \dots, 1)^t$                    [arbitrary]
2. $i = 0$
3. Solve $(A - h^2\Lambda_i I)\, u_{i+1} = Y_i$ for $u_{i+1}$
4. Compute $\Lambda_{i+1} = u_{i+1}^t A u_{i+1} / h^2 u_{i+1}^t u_{i+1}$
   (Rayleigh quotient)
5. Compute $Y_{i+1} = u_{i+1} / (u_{i+1}^t\, u_{i+1})^{\frac{1}{2}}$
   (Normalization of the eigenvector)
6. If $|\Lambda_{i+1} - \Lambda_i| < \epsilon$ STOP else $i = i+1$ and
   GO TO 3.
```

Ostrowski shows that if A is symmetric this process converges cubically to an eigenvalue, i.e.

$$|\Lambda_{i+1} - \Lambda| \approx c|\Lambda_i - \Lambda|^3$$

The shooting methods described in Chapter 3 rely on

Newton's method for convergence and thus exhibit only quadratic convergence to the eigenvalue.

For the remainder of this chapter we assume that the approximate eigenvalue required to start Algorithm 4.2 and all other algorithms to follow is always available. In Chapter 6 we present an algorithm for efficiently computing a good approximation.

4.1.4 Error Estimate and Deferred Correction

Theorem 4.1 showed that by taking h small enough we can get $\boldsymbol{\Lambda}$ arbitrarily close to λ. Additionally, for any particular h we are able to estimate the error in $\boldsymbol{\Lambda}$.

Recalling (4.1.6),

$$A\bar{y} - h^2\lambda\bar{y} = h^2 \quad \bar{\boldsymbol{\tau}}(\bar{y}) \tag{4.1.6}$$

we let $\bar{y} = Y + \Delta y$ and $\lambda = \boldsymbol{\Lambda} + \Delta\lambda$. Substituting into (4.1.6) and multiplying by Y^t, we obtain

$$\begin{aligned} Y^t\{AY - h^2\boldsymbol{\Lambda}Y\} + \{A^tY - h^2\boldsymbol{\Lambda}Y\}^t\Delta y \\ - h^2\Delta\lambda Y^t\bar{y} = h^2Y^t \quad \bar{\boldsymbol{\tau}}(\bar{y}) \end{aligned} \tag{4.1.7}$$

The first term of (4.1.7) is equal to zero by (4.1.4). The second term is also zero since A is symmetric.

Solving for $\Delta\lambda$ we get

$$\Delta\lambda = - Y^t \; \bar{\boldsymbol{\tau}}(\bar{y}) \; / \; Y^t\bar{y}$$

which is an exact expression for the error in $\boldsymbol{\Lambda}$. This expression is not a practical one, however, because it involves $\bar{v}$, the exact solution. If in (4.1.7) we assume that $\Delta\lambda\Delta y$ is negligible then an approximation for $\Delta\lambda$ becomes

$$\Delta\lambda \approx - Y^t \; \bar{\boldsymbol{\tau}}(\bar{y}) \; / \; Y^tY \qquad (4.1.8)$$

The vector $\bar{\boldsymbol{\tau}}(\bar{y})$ is, of course, also a function of the exact solution $\bar{y}$, but $\bar{\boldsymbol{\tau}}(\bar{y})$, too, can be approximated by taking the appropriate differences of the $\{Y_i\}$ or equivalently the $\{Y_i''\}$. A method for approximating the truncation error is presented in Chapter 5. For now we assume only that we have available a method which produces the approximation

$$T_i(Y) = \boldsymbol{\tau}_i(y)\{1 + O(h^r)\}, \quad r \geq 1.$$

Let us define $\bar{T}(Y) = (T_1(Y), T_2(Y), \ldots, T_N(Y) \;)^t$. We can now introduce an algorithm known as deferred correction and made popular by Fox[18]. It enables us to improve upon the matrix eigenvalue and eigenvector and is described as follows.

ALGORITHM 4.3: Let Y and Λ be the solution to the matrix problem (4.1.4).

1. Estimate $\bar{\tau}(\bar{y})$ by computing $\bar{T}(Y)$.
2. Estimate $\Delta\lambda$ by computing $\overline{\Delta\lambda} = -Y^t\bar{T}(Y) / Y^tY$.
3. Let $\bar{\lambda} = \Lambda + \overline{\Delta\lambda}$.
4. Solve $(A - h^2\bar{\lambda}I)\,\bar{\bar{y}} = \bar{T}(Y)$ for $\bar{\bar{y}}$.

The solution $(\bar{\bar{y}},\bar{\lambda})$ is now presumed to be a better approximation to (y,λ). The work by Fischer[16], although it is concerned specifically with a higher order method, implies that if Y were found to be equal to $\bar{y} + O(h^2)$, then $\bar{\bar{y}} = \bar{y} + O(h^4)$. From what follows Theorem 4.1 we know $\Delta\lambda$ to be $O(h^2)$. Thus, one correction by Algorithm 4.3 also yields $\bar{\lambda} = \lambda + O(h^4)$.

Algorithm 4.3 can also be used iteratively by recomputing $\Delta\lambda$ and $T(\bar{\bar{y}})$ and repeating; however, Fox argues that usually only one correction is necessary to get highly accurate results.

Numerical results using the Stormer method are presented at the end of Section 4.2, at which time they are compared with the higher order Numerov method of that section.

4.2 The Numerov Uniform Mesh Method

As the next step toward our adaptive method we introduce a higher order approximation to (4.1.1), (4.1.2). While this method still uses a uniform mesh it possesses some properties different from those of the Stormer method and thus serves to introduce some concepts essential to the adaptive method.

The discretization we use is as follows.

$$- Y_{i-1} + 2Y_i - Y_{i+1} + h^2(f_{i-1}Y_{i-1} + 10f_iY_i + f_{i+1}Y_{i+1})/12 = 0, \quad i=1,2,\dots,N$$

$$Y_0 = Y_{N+1} = 0 \qquad (4.2.1)$$

where $f_i = (q_i - \Lambda)$. This is known as the Numerov method.

When we write (4.2.1) in matrix form we have

$$AY - \Lambda BY = 0 \qquad (4.2.2)$$

where Y is as before, A is the N x N matrix, $A = (a_{ij})$, with

$$\begin{aligned} a_{j,j-1} &= -1 + h^2q_{i-1} / 12 \\ a_{j,j+1} &= -1 + h^2q_{i+1} / 12 \\ a_{jj} &= 2 + 10h^2q_i / 12 \\ a_{ij} &= 0 \quad \text{if } |i-j| > 1 \end{aligned}$$

and B is the N x N matrix, $B = (b_{ij})$, with

$$\begin{aligned} b_{j,j-1} = b_{j,j+1} &= h^2/12 \\ b_{jj} &= 10h^2/12 \\ b_{ij} &= 0 \quad \text{if } |i-j| > 1. \end{aligned}$$

The matrix B is tridiagonal, symmetric, and positive-definite, but, although A is still tridiagonal, we see that it is no longer symmetric. (Figure 4.1)

$$A = \frac{1}{12}\begin{bmatrix} 24+10h^2q_1 & -12+h^2q_2 & & & \\ -12+h^2q_1 & 24+10h^2q_2 & -12+h^2q_3 & & \\ & -12+h^2q_2 & 24+10h^2q_3 & -12+h^2q_4 & \\ & & \cdot & \cdot & \cdot \\ & & & \cdot & \cdot \end{bmatrix}$$

Figure 4.1

The Matrix A

Matrix A possesses, however, another quite useful property. First, we make the following definition.

DEFINITION: (Wilkinson[42]) Let $C = (c_{ij})$ be an $n \times n$, real, tridiagonal matrix defined by $c_{ii} = \alpha_i$, $c_{i,i+1} = \beta_{i+1}$, $c_{i+1,i} = \gamma_{i+1}$, $c_{ij} = 0$ otherwise.
Then C is quasi-symmetric if $\beta_i\gamma_i > 0$ for $i=1,2,\ldots,n$.

A quasi-symmetric matrix can be transformed into a real, symmetric matrix as follows.

Let $d_{11} = 1$, $d_{ii} = (\gamma_1\gamma_2\cdots\gamma_i/\beta_1\beta_2\cdots\beta_i)^{\frac{1}{2}}$, $d_{ij} = 0$ for $i \neq j$. Then $D = (d_{ij})$ is such that $D^{-1}CD = T$ is symmetric with $t_{ii} = \alpha_i$, $t_{i,i+1} = t_{i+1,i} = (\beta_{i+1}\gamma_{i+1})^{\frac{1}{2}}$.

We can now make the following observation about the matrix A.

THEOREM 4.4: For h sufficiently small, A is quasi-symmetric.

Proof: For the theorem to be true we must find an h_0 such that for all $h \leq h_0$ we have

$$a_{i,i+1} \cdot a_{i+1,i} = (-1 + h^2 q_{i+1}/12)(-1 + h^2 q_i/12) > 0. \tag{4.2.3}$$

If we let $Q = \max_j |q_j|$ then by choosing h_0 such that

$$-1 + h_0^2 Q/12 < 0 \quad \text{or} \quad h_0 < (12/Q)^{\frac{1}{2}}$$

each term on the right-hand side of (4.2.3) will be negative for $h \leq h_0$ and the theorem is proved.

One further property of the matrix problem (4.2.2) which we must show is

PROPERTY I: For h sufficiently small the matrix problem (4.2.2) has N real, distinct eigenvalues.

Proof: The reader is referred to Corollary 4.18 of Theorem 4.16 for the proof.

Because the Numerov method is merely a special case of the non-uniform mesh method presented in Section 4.3, several results thoughout the remainder of this section are not proved here but rather as corollaries in Section 4.3.

4.2.1 Truncation Error

The truncation error for the method (4.2.1) is found in the same manner as with the previous method and we find that

$$\tau(x_i) = \frac{1}{240} h^6 y^{(vi)}(\theta), \quad x_{i-1} < \theta < x_{i+1}$$
$$i = 1, 2, \ldots, N$$

Defining $\bar{\tau}(\bar{y}) = (\tau(x_1),\tau(x_2),\ldots,\tau(x_N))^t$ we can write

$$A\bar{y} - \lambda B\bar{y} = \bar{\tau}(\bar{y}) \qquad (4.2.4)$$

4.2.2 Convergence

The task of showing that the matrix eigenvalues converge to the exact eigenvalues is slightly complicated by the quasi-symmetry of A. Keller[25] presents a theorem which states that:

If A is symmetric and B symmetric-positive-definite then for each eigenvalue λ of (4.1.1) and corresponding normalized eigenfunction $y(x)$ there exists an eigenvalue Λ of $B^{-1}A$ such that

$$|\Lambda - \lambda| \leq \|B^{-1}\|\cdot\|\bar{\tau}(\bar{y})\| / \|\bar{y}\|$$

We have shown, however, that the Numerov method produces an A which is quasi-symmetric. The problem of adapting Keller's theorem to accomodate the quasi-symmetry of A appears quite difficult. Instead we prove convergence in a slightly different manner.

The statement of our result is as follows:

PROPERTY II: For the Numerov method, (4.2.1), there exists , for every eigenvalue λ and corresponding eigenfunction $y(x)$ of the differential equation, an eigenvalue Λ of the matrix problem (4.2.2) and a corresponding eigenvector Y such that

$$(\Lambda - \lambda) = O(h^4)$$

and $$Y_i = y(x_i) + O(h^4), \quad i=0,1,\ldots,N+1.$$

Proof: See Corollary 4.20.

We see now, as with the Stormer method, that by taking h sufficiently small we can approximate the exact solution to within an arbitrary tolerance.

4.2.3 Solution of the Matrix Problem

As with the Stormer method of Section 4.1, we want to solve the matrix problem (4.2.2) for only one eigenvalue and eigenvector at a time. We again use Rayleigh quotient iteration.

It is important to note here that the fact that our problem has real eigenvalues is essential to us now. Rayleigh quotient iteration, as defined here, can only converge to a real eigenvalue. Although it was not mentioned at the time, the real eigenvalue property also holds for the Stormer method since all eigenvalues of a symmetric matrix are real.

We present now the algorithm for solving the generalized matrix eigenvalue problem (4.2.2) by Rayleigh quotient iteration.

ALGORITHM 4.5: Let A and B be the real, N x N matrices described above. Let Λ_0 be an approximation to Λ.

```
1. Let $BY_0 = (1,1,\ldots,1)^t$                [arbitrary]
2. $i = 0$
3. Solve $(A - \Lambda_i B)u_{i+1} = BY_i$ for $u_{i+1}$
4. Compute $\Lambda_{i+1} = u_{i+1}^t A u_{i+1} / u_{i+1}^t B u_{i+1}$
```

5. Compute $Y_{i+1} = u_{i+1} / (u_{i+1}^t u_{i+1})^{\frac{1}{2}}$
6. If $|\Lambda_{i+1} - \Lambda_i| < \epsilon$ STOP else $i = i+1$ and GO TO 3.

Ostrowski also discusses this process in his series and shows that, for A symmetric and B symmetric-positive-definite, cubic convergence of Λ_i to Λ is again present. Unfortunately this result does not immediately carry over to the case in which A is quasi-symmetric. However, we show in Section 4.3.6 that because of the interdependence between the matrix problem and the differential equation, it is still possible to obtain cubic convergence under certain conditions.

4.2.4 Error Estimate and Deferred Correction

The derivation of an error expression for $(\Lambda - \lambda)$ is again possible.

Starting with (4.2.4),

$$A\bar{y} - \lambda B\bar{y} = \bar{\tau}(\bar{y}), \qquad (4.2.4)$$

letting $\bar{y} = Y + \Delta y$ and $\lambda = \Lambda + \Delta\lambda$, and substituting into (4.2.4) while multiplying by Y^t we get

$$Y^t A(Y + \Delta y) - (\Lambda + \Delta\lambda) Y^t B(Y + \Delta y) = Y^t \bar{\tau}(\bar{y}).$$

Multiplying out, we have

$$Y^t\{AY - \Lambda BY\} + Y^t\{A\Delta y - \Lambda B\Delta y\} - - \Delta\lambda Y^t BY - \Delta\lambda Y^t B\Delta y = Y^t \bar{\tau}(\bar{y}) \qquad (4.2.5)$$

The first term is zero directly from (4.2.2). The fourth term contains the product $\Delta\lambda\Delta y$ and so can be considered negligible in relation to the other terms.

The second term we can also consider to be negligible by the following reasoning.

$$AY = \Lambda BY$$
$$Y^tA^t = \Lambda Y^tB^t = \Lambda Y^tB$$
$$Y^tA^t\Delta y = \Lambda Y^tB\Delta y$$
$$Y^tA\Delta y - \Lambda Y^tB\Delta y = Y^tA\Delta y - Y^tA^t\Delta y$$
$$\therefore \quad Y^t(A - \Lambda B)\Delta y = Y^t(A - A^t)\Delta y$$

However, the matrix $(A - A^t)$ consists of 0's on the diagonal and terms like $h^2(q_i - q_{i-1})/12$ on the super and subdiagonals.

With $q(x)$ being continuous on $[a,b]$, these terms are $O(h^3)$. Thus, when we solve (4.2.5) for $\Delta\lambda$, eliminating the zero and negligible terms, we have

$$\Delta\lambda \approx -\frac{Y^t \;\bar{\tau}(\bar{y})}{Y^tBY} - \frac{Y^t(A - A^t)\Delta y}{Y^tBY}$$

and we find that the first term is $O(h^4)$ while the second is $O(h^5)$. This leads us to the following error estimate.

$$\Delta\lambda \approx - Y^t \;\bar{\tau}(\bar{y}) \,/\, Y^tBY. \qquad (4.2.6)$$

We note again that to use this estimate we must approximate $\bar{\tau}(\bar{y})$, and such an approximation is presented in Chapter 5. For now we assume that $T_i(Y) = \tau_i(y) \cdot \{1 + O(h^r)\}$, $r \geq 1$, is available.

The error estimate (4.2.6) now makes available the means for correcting the Numerov method in the fashion of Algorithm 4.3.

ALGORITHM 4.6: Let Y and Λ be the solution to the matrix problem (4.2.2).

1. Estimate $\bar{\tau}(\bar{y})$ by computing $\bar{T}(Y)$.
2. Estimate $\Delta\lambda$ by computing $\overline{\Delta\lambda} = -Y^t\bar{T}(Y) / Y^tBY$.
3. Let $\bar{\lambda} = \Lambda + \Delta\lambda$.
4. Solve $(A - \bar{\lambda}B)\bar{\bar{y}} = \bar{T}(Y)$ for $\bar{\bar{y}}$.

The improvement to the eigenvector by making one such correction was shown by Fischer[16] to be from an error of $O(h^4)$ to $O(h^6)$. A similar improvement is experienced in the eigenvalue. (See results below)

4.2.5 Numerical Results

In Tables 4.1 and 4.2 are presented some numerical results for the following problem, known as Weber's equation.

$$y'' + (\lambda - x^2)y = 0$$
$$y(0) = y(1) = 0$$

Table 4.1 shows results for $\lambda_0 \approx 10.1511640305$ while Table 4.2 represents $\lambda_2 \approx 89.1543424562$.

Each table shows the following for N = 8,16,32, 64, and 128.

1. Absolute value of error estimate (4.1.8)
2. $|\Lambda - \lambda|$ for Stormer's method (E_s)
3. E_s/h^2
4. Absolute value of (4.2.6)
5. $|\Lambda - \lambda|$ for the Numerov method (E_n)
6. E_n/h^4
7. $|\bar{\lambda} - \lambda|$ for Stormer's method (E_{sc})
   (Note: this is the corrected eigenvalue.)
8. E_{sc}/h^4
9. $|\bar{\lambda} - \lambda|$ for Numerov method (E_{nc})
10. E_{nc}/h^6

Table 4.1

Results for λ_0

N	|4.1.8|	E_s	E_s/h^2	E_{sc}	E_{sc}/h^4
8	1.237D-1	1.263D-1	8.083	2.654D-3	10.87
16	3.152D-2	3.169D-2	8.113	1.675D-4	10.98
32	7.920D-3	7.929D-3	8.119	1.049D-5	11.00
64	1.982D-3	1.983D-d	8.122	6.561D-7	11.01
128	4.957D-4	4.957D-4	8.123	4.11 D-8	11.03

N	|4.2.6|	E_n	E_n/h^4	E_{nc}	E_{nc}/h^6
8	1.122D-3	1.066D-3	4.366	5.541D-5	14.53
16	6.596D-5	6.628D-5	4.344	3.166D-7	5.312
32	4.130D-6	4.140D-6	4.341	6.3 D-9	6.765
64	2.584D-7	2.600D-7	4.362	1.0 D-10	6.872
128	1.615D-8	2.000D-8	5.369	5.0 D-11	*2199.

* λ_0 correct to all available digits

Table 4.2

Results for λ_2

N	|4.1.8|	E_s	E_s/h^2	E_{sc}	E_{sc}/h^4
8	8.130D0	9.811D0	627.9	1.681D0	6885.
16	2.434D0	2.539D0	650.0	1.681D-1	7556.
32	6.329D-1	6.403D-1	655.7	7.372D-3	7731.
64	1.599D-1	1.604D-1	657.0	4.634D-4	7774.
128	4.010D-2	4.012D-2	657.3	2.901D-5	7787.
N	|4.2.6|	E_n	E_n/h^4	E_{nc}	E_{nc}/h^6
8	6.992D-1	7.488D-1	3067.	4.962D-2	13008.
16	4.877D-2	4.521D-2	2963.	3.562D-3	59760.
32	2.785D-3	2.797D-3	2920.	1.264D-5	13572.
64	1.740D-4	1.744D-3	2926.	4.299D-7	29543.
128	1.089D-5	1.089D-5	2923.	7.2 D-9	31666.

Tables 4.1 and 4.2 illustrate several points. We see the improvement in order from $O(h^2)$ to $O(h^4)$ when the Numerov method is used. We note that both methods experience the expected increase in order when the correction is made. Finally, we note that the error estimates do appear to approach asymptotically the actual error.

4.3 A Non-Uniform Finite-Difference Method

It is the contention of this thesis that a more efficient method of solving the S-L problem is to use finite-difference methods adaptively with a non-uniform mesh. In preparation for such a method we present a non-uniform discretization and describe its application to our problem. Our selection of the particular form of the discretization was influenced by the efficiency of the Numerov method of Section 4.2. We wish to retain the tri-diagonal property of the matrices, the cubic convergence of Rayleigh quotient iteration, and the $O(h^4)$ behavior of the error. As we show in this section, the following form does preserve these properties.

Let $\pi = \{x_i\}_{i=0}^{N+1}$ be a mesh defined on $[a,b]$ with $a = x_0 < x_1 < \ldots < x_N < x_{N+1} = b$. Let $h_i = x_i - x_{i-1}$, $i=1,2,\ldots,N+1$.

We now introduce the following discretization of the differential equation (4.1.1).

$$\alpha_{0i}Y_{i-1} + 2Y_i + \alpha_{2i}Y_{i+1} + \beta_{0i}Y''_{i-1} + \beta_{1i}Y''_i + \beta_{2i}Y''_{i+1} = 0, \quad i=1,2,\ldots,N \qquad (4.3.1)$$

where α_{0i}, α_{2i}, β_{0i}, β_{1i}, and β_{2i} are real coefficients to be determined and $Y''_i = (q(x_i) - \Lambda)Y_i$.

If we now require that this approximation be exact for $y(x) = 1, x, x^2, x^3$, and x^4 we get the following set of equations:

$y(x) = 1,\ y''(x) = 0$

$$\alpha_{0i} + 2 + \alpha_{2i} = 0$$

$y(x) = x,\ y''(x) = 0$

$$\alpha_{0i}x_{i-1} + 2x_i + \alpha_{2i}x_{i+1} = 0$$

$y(x) = x^2,\ y''(x) = 2$

$$\alpha_{0i}x_{i-1}^2 + 2x_i^2 + \alpha_{2i}x_{i+1}^2 + 2\beta_{0i} + 2\beta_{1i} + 2\beta_{2i} = 0$$

$y(x) = x^3,\ y''(x) = 6x$

$$\alpha_{0i}x_{i-1}^3 + 2x_i^3 + \alpha_{2i}x_{i+1}^3 + 6\beta_{0i}x_{i-1} + 6\beta_{1i}x_i + 6\beta_{2i}x_{i+1} = 0$$

$y(x) = x^4,\ y''(x) = 12x^2$

$$\alpha_{0i}x_{i-1}^4 + 2x_i^4 + \alpha_{2i}x_{i+1}^4 + 12\beta_{0i}x_{i-1}^2 + 12\beta_{1i}x_i^2 + 12\beta_{2i}x_{i+1}^2 = 0$$

Solving the five equations simultaneously and replacing the x_i's with appropriate h_i's, we get

$$\alpha_{0i} = -2h_{i+1} / (h_i + h_{i+1})$$
$$\alpha_{2i} = -2h_i / (h_i + h_{i+1})$$
$$\beta_{0i} = -h_{i+1}(h_{i+1}^2 - h_ih_{i+1} - h_i^2) / 6(h_i + h_{i+1})$$
$$\beta_{1i} = (h_i^2 + 3h_ih_{i+1} + h_{i+1}^2) / 6$$
$$\beta_{2i} = -h_i(h_i^2 - h_ih_{i+1} - h_{i+1}^2) / 6(h_i + h_{i+1})$$

Thus the difference equation is, in general, different at each point x_i.

These coefficients are similar to those used by Brown[7] to solve non-linear boundary value problems. He does not, however, consider eigenvalue problems.

As with the previous discretizations we can now write equations (4.3.1) in matrix form to get

$$(A - \Lambda B)\ Y = 0 \qquad (4.3.2)$$

where $A = (a_{ij})$ and $B = (b_{ij})$ are N x N real, tridiagonal matrices and $Y = (Y_i)$ is a real column vector of length N with

$$a_{ij} = \begin{cases} 2 + \beta_{1i} q_i & \text{if } j = i \\ \alpha_{0i} + \beta_{0i} q_{i-1} & \text{if } j = i-1 \\ \alpha_{2i} + \beta_{2i} q_{i+1} & \text{if } j = i+1 \\ 0 & \text{otherwise} \end{cases}$$

and

$$b_{ij} = \begin{cases} \beta_{1i} & \text{if } j = i \\ \beta_{0i} & \text{if } j = i-1 \\ \beta_{2i} & \text{if } j = i+1 \\ 0 & \text{otherwise} \end{cases}$$

The problem now becomes that of finding the eigenvalues, Λ, and eigenvectors, Y, of the matrix problem (4.3.2).

We note that the symmetry and positive-definite properties of A and B are no longer present. Only the tri-diagonal property remains. If some further simplifying properties were not available for A and B, a general algorithm such as the QZ algorithm (see Stewart[39], Chapter 7) would be required to solve the problem. We are, however, able to show several interesting properties of the matrix problem, each of which serves to simplify the solution process.

4.3.1 Properties of the Matrices A and B

First, we note two obvious facts about A and B. Since h_i is always positive it follows that:

PROPERTY 4.7: α_{0i} and α_{2i} are negative for all i.

PROPERTY 4.8: β_{1i} is positive for all i.

These properties are immediately obtained from the expressions for α_{0i}, α_{2i}, and β_{1i}.

In the paper by Brown, mentioned above, he requires that β_{0i} and β_{2i} always be positive. This is essential to his proof of the convergence of Newton's method for solving his non-linear problems.

With the following lemma we show the severe restrictions this places on the relative sizes of adjacent intervals, h_i and h_{i+1}.

Let $h_{i+1} = a \cdot h_i$ where $a > 0$.

LEMMA 4.9:

i) $\beta_{0i} = \beta_{2i} = 1/12$ when $h_i = h_{i+1}$ $(a = 1)$.

ii) Both β_{0i} and β_{2i} are > 0 if and only if

$$\frac{\sqrt{5}-1}{2} < a < \frac{\sqrt{5}+1}{2}$$

iii) $\beta_{0i}\beta_{2i} > 0$ if and only if $\beta_{0i} > 0$ and $\beta_{2i} > 0$.

Proof:

i) This is immediately obvious by substituting $h_i = h_{i+1}$ into the expressions for β_{0i} and β_{2i}. Also here we note that $\beta_{1i} = 10/12$ when $h_i = h_{i+1}$. These coefficients are of course what we expect since when $h_i = h_{i+1}$ the discretization is simply the Numerov method of Section 4.2.

ii) $$\beta_{0i} = -h_{i+1}(h_{i+1}^2 - h_i h_{i+1} - h_i^2) / 6(h_i + h_{i+1})$$
$$= (-a^2 + a + 1)h_{i+1} / 6(h_i + h_{i+1})h_i^2$$

So $\beta_{0i} > 0$ if and only if $-a^2 + a + 1 > 0$ which occurs if and only if

$$\frac{-\sqrt{5} + 1}{2} < a < \frac{\sqrt{5} + 1}{2} .$$

Similarly $\beta_{2i} > 0$ if and only if

$$a < \frac{-\sqrt{5} - 1}{2} \quad \text{or} \quad \frac{\sqrt{5} - 1}{2} < a.$$

Thus, both β_{0i} and $\beta_{2i} > 0$ if and only if

$$\frac{\sqrt{5} - 1}{2} < a < \frac{\sqrt{5} + 1}{2}$$

or $\qquad .618 < a < 1.618$ (approximately).

iii) By steps similar to those of ii) we find that $\beta_{0i} < 0$ if and only if

$$a < \frac{-\sqrt{5} + 1}{2} \quad \text{or} \quad \frac{\sqrt{5} + 1}{2} < a$$

and that $\beta_{2i} < 0$ if and only if

$$\frac{-\sqrt{5} - 1}{2} < a < \frac{\sqrt{5} - 1}{2} .$$

There is no positive value of a which can satisfy both and so we conclude that β_{0i} and β_{2i} cannot both be negative.

$\therefore$ iii) follows from ii).

The significance of this result is as follows. If we were to require, as Brown did, that the entries of B all be positive, then we must restrict the ratio of adjacent steps to that of ii). This is so restrictive that steps cannot even differ by a factor of 2.

If we lift the restriction and require only that steps be changed by an integer factor (i.e. $a = n$ or $1/n$, n a positive integer) we get the following corollary.

COROLLARY 4.10: If adjacent step sizes always differ by an integer factor then either $\beta_{0i} = \beta_{2i} = 1/12$ or $\beta_{0i}\beta_{2i} < 0$.

We see in Section 5.3 that this condition is automatically met with our method of refining the mesh.

4.3.1.1 Quasi-symmetry of A

With the above properties available it is now possible to show that under certain conditions the matrix A is quasi-symmetric, as was the case with the Numerov method.

Letting $h = \max_{1 \le i \le N+1} h_i$ we have the following:

THEOREM 4.11: For h sufficiently small, A is quasi-symmetric.

Proof: For A to be quasi-symmetric we need to find an h_0 such that for all $h \le h_0$

$$(\alpha_{0i} + \beta_{0i}q_{i-1})(\alpha_{2,i-1} + \beta_{2,i-1}q_i) > 0 \text{ for all } i.$$

Let $h_i = ph$ and $h_{i+1} = rh$ where $0 < p \leq 1$ and $0 < r \leq 1$.

We first show that for h sufficiently small $\alpha_{0i} + \beta_{0i}q_{i-1} < 0$ or $\beta_{0i}q_{i-1} < -\alpha_{0i}$.

From Property 4.7 it is known that $\alpha_{0i} < 0$. We also note that

$$\begin{aligned}\beta_{0i} &= -h_{i+1}(h_{i+1}^2 - h_ih_{i+1} - h_i^2) / 6(h_i + h_{i+1}) \\ &= \alpha_{0i}(h_{i+1}^2 - h_ih_{i+1} - h_i^2) / 12 \\ &= -\alpha_{0i}(h^2/12)(p^2 + pr - r^2)\end{aligned}$$

So we want

$$-\alpha_{0i}(h^2/12)(p^2 + pr - r^2)q_{i-1} < -\alpha_{0i}$$

or dividing by $-\alpha_{0i}/12$, which is positive, we get

$$h^2(p^2 + pr - r^2)q_{i-1} < 12. \qquad (4.3.3)$$

Letting $z(p,r) = p^2 + pr - r^2$, we wish to find $\max_R |z(p,r)|$ where $R = \{(p,r) \mid 0 < p \leq 1,\ 0 < r \leq 1\}$.

The critical point occurs where

$$\partial z/\partial p = \partial z/\partial r = 0$$

or where

$$2p + r = 0$$
$$p - 2r = 0$$

or at $(p,r) = (0,0)$. $(z = 0)$

Hence, the maximum occurs on a boundary, namely at

$(p,r) = (1,\frac{1}{2})$ where $z = 5/4$.

$\therefore \max_{R} |z(p,r)| = 5/4$.

If we also define $Q = \max_{a \le x \le b} |q(x)|$, we find that (4.3.3) holds if

$$h < (48/5Q)^{\frac{1}{2}}.$$

We also need $\beta_{2,i-1} q_i < -\alpha_{2i}$ which leads to the same restriction on h.

Thus letting $h_0 = (48/5Q)^{\frac{1}{2}}$ we see that for $h \le h_0$ A is quasi-symmetric.

4.3.1.2 Properties of B

We have already noted that the matrix B is no longer symmetric, positive-definite, however, B does still possess several of the important properties of a positive-definite matrix and we present them here.

The first such property, although important in itself, is mainly used to prove several facts, and so is introduced here as a lemma.

LEMMA 4.12: B is strictly diagonally dominant.

Proof: We must show that

$$|\beta_{1i}| > |\beta_{0i}| + |\beta_{2i}| \quad \text{for } i=1,2,\ldots,N.$$

$$|\beta_{0i}| + |\beta_{2i}| = [\,|h_{i+1}^3 - h_i h_{i+1}^2 - h_i^2 h_{i+1}| + |h_i^3 - h_i^2 h_{i+1} - h_i h_{i+1}^2|\,]/6(h_i + h_{i+1})$$

$$\leq [h_i^3 + 2h_i^2 h_{i+1} + 2h_i h_{i+1}^2 + h_{i+1}^3]/6(h_i + h_{i+1})$$

$$< [h_i^3 + 3h_i^2 h_{i+1} + 3h_i h_{i+1}^2 + h_{i+1}^3]/6(h_i + h_{i+1})$$

$$= [(h_i + h_{i+1})^3] / 6(h_i + h_{i+1})$$

$$= [h_i^2 + 2h_i h_{i+1} + h_{i+1}^2] / 6$$

$$< [h_i^2 + 3h_i h_{i+1} + h_{i+1}^2] / 6$$

$$= \beta_{1i} = |\beta_{1i}|.$$

The next result we need for our ultimate statement about the eigenvalues of (4.3.2) is one about the determinants of B and its leading principal sub-matrices. The desired result appears as a corollary to the following theorem.

THEOREM 4.13: If B is a real n x n matrix and is strictly diagonally dominant with positive diagonal elements, then det(B) > 0.

Proof: By a theorem of Gerschgorin, taken here from Varga[41], we know that the eigenvalues, $\{\lambda_i\}$, of a strictly diagonally dominant matrix with positive diagonal entries satisfy

$$\mathrm{Re}(\lambda_i) > 0, \ 1 \leq i \leq n.$$

This, of course, implies that all real eigenvalues are positive.

Also it is known (see for example Stewart[39],

p. 269) that the complex eigenvalues of a real matrix occur in conjugate pairs and that the product of a number and its complex conjugate is a positive real number.

Thus,

$$\prod_{j=1}^{n} \lambda_j > 0. \tag{4.3.4}$$

But

$$\prod_{j=1}^{n} \lambda_j = \det(B)$$

as the following argument shows.

Let $p(\lambda)$ be the characteristic polynomial of B, i.e. $p(\lambda) = \det(B - \lambda I)$. We know that $p(\lambda)$ can be factored into

$$\begin{aligned} p(\lambda) &= (-1)^n(\lambda - \lambda_1)(\lambda - \lambda_2) \ldots (\lambda - \lambda_n) \\ &= \det\,(B - \lambda I) \end{aligned}$$

Letting $\lambda = 0$ in this identity, we get

$$(-1)^n(-1)^n \prod_{j=1}^{n} \lambda_j = \det\,(B). \tag{4.3.5}$$

And so combining (4.3.4) and (4.3.5) we see that

$$\det\,(B) > 0.$$

This theorem can also be shown to be an immediate corollary to the following theorem by G. B. Price[34], stated here without proof.

THEOREM 4.14: Let $A = (a_{ij})$ be an n x n matrix with real elements such that

$$a_{ii} > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|$$

then

$$0 < m_1 m_2 \ldots m_n \leq \det (A) \leq M_1 M_2 \ldots M_n$$

where

$$m_i = |a_{ii}| - \sum_{j=i+1}^{n} |a_{ij}|$$

$$M_i = |a_{ii}| + \sum_{j=i+1}^{n} |a_{ij}|$$

If we now define B_i to be the leading principal submatrix of B of order i, we get the following:

COROLLARY 4.15: For the matrix problem (4.3.2) $\det (B_i) > 0$, $i=1,2,\ldots,N$.

Proof: Each B_i is a real i x i matrix. From Lemma 4.12, each B_i is strictly diagonally dominant. From Property 4.8, each B_i has positive diagonal elements. $\therefore$ By Theorem 4.13, $\det (B_i) > 0$, $i=1,2,\ldots,N$.

It is possible to develop a 3-term recurrence relationship for $\det (B_i)$. Because such a relationship is needed in a later proof, we introduce it here.

Let us define $\det(B_0) = 1$ and $\det(B_1) = \beta_{11}$.

Recalling that B, and thus B_i, are tridiagonal we see that $\det(B_i)$ can be computed by expanding B_i about row i to yield

$$\det(B_i) = \beta_{1i}\det(B_{i-1}) - \beta_{0i}\beta_{2,i-1}\det(B_{i-2}) \qquad i=2,3,\ldots,N \qquad (4.3.6)$$

Expression (4.3.6) is the 3-term recurrence relationship we need.

4.3.2 Truncation Error

The next property of our discretization we wish to look at is the truncation error. It can be computed by first noting that

$$\begin{aligned}\tau(x_i) &= \alpha_{0i}y(x_{i-1}) + 2y(x_i) + \alpha_{2i}y(x_{i+1}) + \\ &\quad \beta_{0i}y''(x_{i-1}) + \beta_{1i}y''(x_i) + \beta_{2i}y''(x_{i+1}) \qquad (4.3.7)\\ &= -2h_{i+1}y(x_i - h_i)/(h_i + h_{i+1}) + 2y(x_i)\\ &\quad -2h_i y(x_i + h_{i+1})/(h_i + h_{i+1})\\ &\quad -(h_{i+1}^3 - h_i h_{i+1}^2 - h_i^2 h_{i+1})y''(x_i - h_i)/6(h_i+h_{i+1})\\ &\quad +(h_i^2 + 3h_i h_{i+1} + h_{i+1}^2)y''(x_i)/6\\ &\quad -(h_i^3 - h_i^2 h_{i+1} - h_i h_{i+1}^2)y''(x_i + h_{i+1})/6(h_i+h_{i+1})\end{aligned}$$

If we now expand $y(x_i - h_i)$, $y(x_i + h_{i+1})$, $y''(x_i - h_i)$, and $y''(x_i + h_{i+1})$ in terms of Taylor's series and then combine like derivatives of y, we get

$$\begin{aligned}\tau(x_i) = {} & \frac{h_i h_{i+1}}{180}\left[2h_{i+1}^3 + 3h_i h_{i+1}^2 - 3h_i^2 h_{i+1} - 2h_i^3\right] y^{(5)}(x_i) \\ & + \frac{h_i h_{i+1}}{720}\left[3h_{i+1}^4 + 2h_i h_{i+1}^3 - 7h_i^2 h_{i+1}^2 + 2h_i^3 h_{i+1} \right. \\ & \left. \qquad + 3h_i^4\right] y^{(6)}(x_i) \\ & + \frac{h_i h_{i+1}}{5040}\left[5h_{i+1}^5 + 2h_i h_{i+1}^4 - 9h_i^2 h_{i+1}^3 + 9h_i^3 h_{i+1}^2 \right. \\ & \left. \qquad - 2h_i^4 h_{i+1} - 5h_i^5\right] y^{(7)}(x_i) \\ & + O(h^8) \qquad (4.3.8)\end{aligned}$$

We should note that when $h_i = h_{i+1}$ the first and third terms of (4.3.8) are zero while the second term is $h^6 y^{(6)}(x_i)/240$ and thus in agreement with the Numerov method. However, when $h_i \neq h_{i+1}$ the truncation error is of order 5 and, in addition, contains a 7th order term.

In Chapter 5 we describe a way to approximate $\tau(x_i)$ once we have computed Y, the approximate eigenfunction.

As we did with the other discretizations, we can now introduce the matrix equation

$$(A - \lambda B)\,\bar{y} = \bar{\tau}(\bar{y}) \qquad (4.3.9)$$

where λ is an exact eigenvalue, $\bar{y}$ is an exact, discretized eigenfunction and

$$\bar{\tau}(\bar{y}) = (\, \tau(x_1), \tau(x_2), \ldots, \tau(x_N)\,)^t.$$

4.3.3 <u>Properties of the Eigenvalues</u>

We recall now, from Chapter 2, that the differential equation we are approximating has an infinite number of real, distinct eigenvalues. Because of the special symmetry property of the matrix in the Stormer method, we knew that the matrix problem also had real, distinct eigenvalues.

If we expect our non-uniform discretization to be a good approximation then we should expect its eigenvalues to be real and distinct. Certainly if we hope to use the Rayleigh quotient iteration to find the eigenvalues and eigenvectors this property is essential.

We are prepared now to state the following:

<u>THEOREM 4.16</u>: For h sufficiently small the N x N matrix problem (4.3.2) has N real, distinct eigenvalues.

Before proving this theorem it is helpful to define the following sequence of polynomials.

$$\begin{aligned}
p_0(\lambda) &= 1 \\
p_1(\lambda) &= \{\beta_{11}(\lambda - q_1) - 2\}/\det(B_1) \\
&\;\;\vdots \\
p_i(\lambda) &= \{[\beta_{1i}(\lambda - q_i) - 2]\det(B_{i-1})p_{i-1}(\lambda) \\
&\quad -[\beta_{2,i-1}(\lambda - q_i) - \alpha_{2,i-1}][\beta_{0i}(\lambda - q_{i-1}) - \alpha_{0i}] \\
&\qquad \det(B_{i-2})p_{i-2}(\lambda)\}/\det(B_i) \\
&\qquad i=2,3,\ldots,N \qquad (4.3.11)
\end{aligned}$$

where $B_0 = 1$ and the B_i, $i=1,2,\ldots,N$, are the leading principal submatrices of B of order i.

LEMMA 4.17: The coefficient of λ^i in $p_i(\lambda)$ is +1 for $i=0,1,\ldots,N$.

Proof of Lemma 4.17: If we note that $\det(B_1) = \beta_{11}$ then the lemma is true for $i=0,1$.

Assume the lemma is true for $i=0,1,\ldots,j-1$. From (4.3.11), the coefficient of λ^j in $p_j(\lambda)$ is

$$\{\beta_{1j}\det(B_{j-1}) - \beta_{0j}\beta_{2,j-1}\det(B_{j-2})\}/\det(B_j)$$

but from relationship (4.3.6) for $\det(B_j)$ we see that this coefficient is equal to +1.

Now we note that the N roots of $p_N(\lambda) = 0$ are exactly the N eigenvalues of (4.3.2). This is obvious if we note that the numerator of $p_N(\lambda)$,

$$[\beta_{1N}(\lambda - q_N) - 2]\det(B_{N-1})p_{N-1}(\lambda) - [\beta_{2,N-1}(\lambda - q_N) - \alpha_{2,N-1}]$$
$$[\beta_{0N}(\lambda - q_{N-1}) - \alpha_{0N}]\det(B_{N-2})p_{N-2}(\lambda),$$

is actually a 3-term recurrence relationship like (4.3.6) but for $\det(\lambda B - A)_N$, or more simply $\det(\lambda B - A)$.

Thus the roots of $p_N(\lambda) = 0$ are exactly the roots of $\det(\lambda B - A) = 0$, which are in turn the eigenvalues of (4.3.2). Thus the proof of Theorem 4.16 reduces to showing that $p_N(\lambda) = 0$ has N real, distinct roots.

Proof of Theorem 4.16: The proof will be by induction. We will show that $p_{i+1}(\lambda) = 0$ has i+1 real roots and that they are separated by the roots of $p_i(\lambda) = 0$.

That is, between any 2 consecutive roots of $p_{i+1}(\lambda)$ there lies exactly 1 root of $p_i(\lambda)$.

Let i=0. Obviously $p_0(\lambda) = 1 = 0$ has no real roots. Also, $p_1(\lambda) = \{\beta_{11}(\lambda - q_1) - 2\}/\det(B_1) = 0$ has 1 real root at $(2 + \beta_{11}q_1)/\beta_{11}$. The separation requirement is satisfied trivially.

Assume now that $p_i(\lambda) = 0$ has i real roots $s_1, s_2, \ldots, s_i$ separated by the i-1 real roots $r_1, r_2, \ldots, r_{i-1}$ of $p_{i-1}(\lambda) = 0$. i.e. $s_1 < r_1 < s_2 < r_2 < \ldots < r_{i-1} < s_i$.

Then by Lemma 4.17 we know that

$$p_i(\lambda) = (\lambda - s_1)(\lambda - s_2) \ldots (\lambda - s_i)$$
$$p_{i-1}(\lambda) = (\lambda - r_1)(\lambda - r_2) \ldots (\lambda - r_{i-1})$$

Thus

$$p_{i+1}(\lambda) = \{[\beta_{1,i+1}(\lambda - q_{i+1}) - 2]\det(B_i)\cdot (\lambda - s_1) \ldots (\lambda - s_i) - [\beta_{2i}(\lambda - q_{i+1}) - \alpha_{2i}][\beta_{0,i+1}(\lambda - q_i) - \alpha_{0,i+1}]\det(B_{i-1})\cdot (\lambda - r_1)\ldots((\lambda - r_{i-1})\}/\det(B_{i+1}) \tag{4.3.12}$$

If we now let s_j and s_{j+1} be any 2 consecutive roots pf $p_i(\lambda) = 0$,we see that $p_{i+1}(s_j)$ and $p_{i+1}(s_{j+1})$ will be of opposite sign, provided the factors $\beta_{2i}(s_j - q_{i+1}) - \alpha_{2i}$ and $\beta_{0,i+1}(s_j - q_i) - \alpha_{0,i+1}$ are of the same sign. This is adequate because $(s_j - r_1)(s_j - r_2)\ldots(s_j - r_{i-1})$ and $(s_{j+1} - r_1)(s_{j+1} - r_2)\ldots(s_{j+1} - r_{i-1})$ are by definition of different signs.

The 2 terms in question can be assured to be of opposite sign by choosing h sufficiently small as we did in the proof of Theorem 4.11.

We now know that $p_{i+1}(\lambda)$ changes sign between s_j and s_{j+1} and therefore has a real root in each such interval.

We also know from (4.3.12) that since s_i is larger than all the r_j, $p_{i+1}(s_i)$ will be negative when h has been chosen sufficiently small. However, Lemma 4.17 implies that $p_{i+1}(\lambda) \to \infty$ as $\lambda \to \infty$ and so $p_{i+1}(\lambda)$ must have a real root greater than s_i.

Similarly, we know from (4.3.12) that $p_{i+1}(s_1)$ has sign $(-1)(-1)^{i-1} = (-1)^i$. However, again from Lemma 4.17, $p_{i+1}(\lambda) \to (-1)^{i+1}\cdot\infty$ as $\lambda \to -\infty$ and so $p_{i+1}(\lambda)$ has a real root less than s_1.

Combining these facts about roots we find that $p_{i+1}(\lambda) = 0$ has i+1 real roots separated by the i roots of $p_i(\lambda) = 0$.

This concludes the proof of Theorem 4.16.

Since the Numerov method can be viewed as just a special case of our non-uniform method, we have immediately

COROLLARY 4.18: For h sufficiently small, the matrix problem (4.2.2) has N real, distinct eigenvalues.

In order to insure the practicality of our non-uniform method we must be able to compute the upper bound on h required by Theorem 4.16. We must also be able to show that such a restriction is not excessive for most problems. Here we develop an expression for such an upper bound.

From Theorem 4.16 we know that h must be small enough to ensure that

$$\beta_{2i}(s_j - q_{i+1}) - \alpha_{2i} > 0 \qquad (4.3.13)$$

for all α, β, s_j, and q_{i+1} and similarly we must have

$$\beta_{0,i+1}(s_j - q_i) - \alpha_{0,i+1} > 0. \qquad (4.3.14)$$

We note that conditions (4.3.13) and (4.3.14) are sufficient to ensure that the sequence (4.3.11) is a Sturm sequence. If we let $Q = \max\limits_{a \leq x \leq b} |q(x)|$ and if we note that $s_j \geq \lambda_0$ and thus let $L = |\lambda_0|$ then by reasoning exactly like that of Theorem 4.11 we find that

$$h < \{48 / 5(Q + L)\}^{\frac{1}{2}} \qquad (4.3.15)$$

is sufficiently small.

It is important to note that while the condition (4.3.15) is sufficient it is certainly not necessary to ensure real eigenvalues. What is necessary is that the expressions (4.3.13) and (4.3.14) have the same sign. This can occur in several other ways.

If $\beta_{2i}(s_j - q_{i+1}) < 0$ and $\beta_{0,i+1}(s_j - q_i) < 0$ then h_i can be of any size, and thus may be very large. If q_i is very small in some subregion of $[a,b]$ then h_i can be larger than (4.3.15) indicates since it uses a global maximum, Q.

We mention these possibilities here because we find in Chapter 7 that excellent results are sometimes obtained even though (4.3.15) may be violated. It is reasonable to state then that (4.3.15) is a sufficient bound but that in practical situations h may be much larger.

4.3.4 Convergence

As we did with the uniform discretizations, we must now consider the question: How accurately can we approximate an eigenvalue and eigenfunction of the differential equation with our matrix solution?

Our next result establishes convergence of the matrix eigenvalue and eigenvector to the eigenvalue and eigenfunction of the differential equation. To get such a result we appeal to the theory of multistep methods for solving ordinary differential equations.

We first introduce some new notation.

Let Λ be the matrix eigenvalue,

$Y_j(\Lambda)$	the jth component of the matrix eigenvector,
λ_j	the differential equation eigenvalue,
$Y_j(\lambda_i)$	the jth component of a vector obtained by using (4.3.1) as a multistep initial value method as follows:

- Let $Y_0(\lambda_i) = 0$ and $Y_1(\lambda_i) = 1$ (arbitrary)
- Integrate outward from a to b using the

relationship (4.3.1) as a multistep method with λ_i an exact eigenvalue to get $Y_j(\lambda_i)$, $j=2,3,\ldots,N$.

$h = \max_i(h_i)$,

$y(x_i)$ the exact solution of the differential equation at x_i normalized such that $y(x_1) = 1$.

We now assume that x_j, $j \geq 1$, is the first point at which the step size changes, i.e. $h_1 = h_2 = \ldots = h_j \neq h_{j+1}$. Such a situation enables us to use a well known theorem of multistep methods (see for example Gear[20], p.190, Theorem 10.8) to state that both

$$Y_{j-1}(\lambda_i) = y(x_{j-1}) + O(h^4)$$

and

$$Y_j(\lambda_i) = y(x_j) + O(h^4).$$

We can now use (4.3.1) to compute $Y_{j+1}(\lambda_i)$ and we find

$$\begin{aligned} Y_{j+1}(\lambda_i) &= \{-\alpha_{0j}Y_{j-1}(\lambda_i) - 2Y_j(\lambda_i) - \beta_{0j}Y''_{j-1}(\lambda_i) \\ &\quad -\beta_{1j}Y''_j(\lambda_i)\} / \{\alpha_{2j} + \beta_{2j}(q_{i+1} - \lambda_i)\} \\ &= \{-\alpha_{0j}[y(x_{j-1}) + O(h^4)] - 2[y(x_j) + O(h^4)] \\ &\quad -\beta_{0j}(q_{j-1} - \lambda_i)[y(x_{j-1}) + O(h^4)] \\ &\quad -\beta_{1j}(q_j - \lambda_i)[y(x_j) + O(h^4)]\} / \\ &\quad \{\alpha_{2j} + \beta_{2j}(q_{j+1} - \lambda_i)\} \end{aligned}$$

but from (4.3.7) we have

$$y(x_{j+1}) = \{ \tau(x_{j+1}) - \alpha_{0j}y(x_{j-1}) - 2y(x_j) - \beta_{0j}y''(x_{j-1}) - \beta_{1j}y''(x_j)\} / \{\alpha_{2j} + \beta_{2j}(q_{j-1} - \lambda_i)\}$$

and so subtracting these equations we get

$$\begin{aligned} Y_{j+1}(\lambda_i) &= \tau(x_j) / \{\alpha_{2j} + \beta_{2j}(q_{j+1} - \lambda_i)\} + y(x_{j+1}) + O(h^4) \\ &= y(x_{j+1}) + O(h^4) \end{aligned}$$

since $\tau(x_j) = O(h^5)$ and $\alpha_{2j} = O(1)$.

Using the numbers $Y_j(\lambda_i)$ and $Y_{j+1}(\lambda_i)$ as starting values we can again use our recurrence relationship (4.3.1) as a multistep method to integrate to the next step size change (say at x_k). Since the errors in the starting values were $O(h^4)$, the theorem mentioned above guarantees us that

$$Y_k(\lambda_i) = y(x_k) + O(h^4).$$

This process can be continued across the entire finite interval $[a,b]$ so that finally

$$Y_{N+1}(\lambda_i) = y(x_{N+1}) + O(h^4).$$

This argument is only valid, however, if the number of step size changes is finite. As $h \to 0$ it is theoretically possible that step size changes could occur at an unbounded number of points. In practice we find that large intervals of uniform step sizes tend to develop in the adaptive process and thus, bounding the number of step size changes in theory will not be restrictive. We therefore introduce the following definition and assume from this point on that our method can be so described.

<u>DEFINITION</u>: A finite-difference method is <u>uniform almost everywhere</u> if there exists a constant $K \geq 0$ such that as $h \to 0$ the number of points x_j where $h_j \neq h_{j+1}$ remains less than or equal to K.

We now consider $Y_{N+1}(\Lambda)$ and note that:

1. $Y_{N+1}(\Lambda) = 0$
2. $Y_{N+1}(\Lambda) = Y_{N+1}(\lambda_i) + (\Lambda - \lambda_i)(\partial Y_{N+1}(\lambda_i)/\partial\lambda) + O(\Lambda - \lambda_i)^2$
$= Y_{N+1}(\lambda_i) + (\Lambda - \lambda_i)z_{N+1} + O(\Lambda - \lambda_i)^2$

where $z_{N+1} = \partial Y_{N+1}/\partial\lambda$ evaluated at λ_i.

We must now look at this new sequence of numbers $\{z_j\}$. Taking the partial derivative of (4.3.1) with respect to λ and evaluating at λ_i yields

$$\alpha_{0i}z_{j-1} + 2z_j + \alpha_{2j}z_{j+1} + \{\beta_{0j}[(q_{j-1} - \lambda_i)z_{j-1} - y_{j-1}] + \beta_{1j}[(q_j - \lambda_i)z_j - y_j] + \beta_{2j}[(q_{j+1} - \lambda_i)z_{j+1} - y_{j+1}]\} = 0 \qquad (4.3.16)$$

Writing (4.3.16) at i=1,2,...,N we get the matrix equation

$$(A - \lambda_i B)\,\bar{z} = B\bar{y} \qquad (4.3.17)$$

In addition, since $Y_0(\Lambda) = 0$ and $Y_1(\Lambda) = 1$, we know that

$z_0 = z_1 = 0$.

Thus $\bar{z}$ is the numerical solution by our method of the differential equation

$$z'' = (q(x) - \lambda_i)\, z - y$$
$$z(0) = z(x_1) = 0$$

which has $z(x,\lambda_i) = \partial y(\lambda_i)/\partial\lambda$ as its solution.

We know from 1. and 2. above that

$$\begin{aligned} 0 = Y_{N+1}(\Lambda) &= Y_{N+1}(\lambda_i) + (\Lambda - \lambda_i) z_{N+1} + O(\Lambda - \lambda_i)^2 \\ &= y(x_{N+1}) + O(h^4) + (\Lambda - \lambda_i) z_{N+1} + O(\Lambda - \lambda_i)^2 \end{aligned}$$

but $y(x_{N+1}) = 0$ from the boundary conditions

$$\therefore\ (\Lambda - \lambda_i) z_{N+1} + O(\Lambda - \lambda_i)^2 = O(h^4).$$

Because the discretization is assumed to be uniform almost everywhere, we know that as $h \to 0$ the solution $\bar{z}$, of (4.3.17), will approach the actual solution z, and thus z_{N+1} will approach $z(x_{N+1},\lambda_i)$. Define $f(\lambda) = y(x_{N+1},\lambda)$ as the value of $y(x_{N+1})$ when (4.1.1) is integrated outward from $y(0) = 0$, $y(x_1) = 1$, using λ as the eigenvalue. We know that when $\lambda = \lambda_i$, an exact eigenvalue, $y(x_{N+1},\lambda_i) = 0$. If $z(x_{N+1},\lambda_i)$ were also zero then λ_i would be a double root of $f(\lambda)$, and λ_i would be an eigenvalue of multiplicity 2 which violates a property of the S-L problem. Thus if we assume that as $h \to 0$, $O(\Lambda - \lambda_i)^2$ becomes negligible compared to $(\Lambda - \lambda_i)$ we find that

$$(\Lambda - \lambda_i) = O(h^4).$$

Now since

$$\begin{aligned} Y_j(\Lambda) &= Y_j(\lambda_i) + (\Lambda - \lambda_i)z_j \\ &= y(x_j) + O(h^4) + O(h^4) \end{aligned}$$

we also have

$$Y_j(\Lambda) = y(x_j) + O(h^4).$$

We have thus proved the following:

THEOREM 4.19: If the finite-difference method (4.3.1) is assumed to be uniform almost everywhere, then for every eigenvalue, λ, of the differential equation (4.1.1) there exists an eigenvalue, Λ, of the matrix problem (4.3.2) and a corresponding eigenvector, Y, such that

$$(\Lambda - \lambda) = O(h^4)$$

and $\quad Y_i(\Lambda) = y(x_i) + O(h^4), \quad i=0,1,\ldots,N+1.$

We also have as an immediate corollary,

COROLLARY 4.20: Theorem 4.19 also holds for the matrix problem (4.2.2) on the uniform mesh.

4.3.5 Error Estimate

In addition to the general convergence results of Theorem 4.19, it is possible to derive an exact expression for $\Delta\lambda = \lambda - \Lambda$.

First, we need to establish the fact that for any given Λ and for h sufficiently small the matrix $A - \Lambda B$ is quasi-symmetric. For this to be true it must be shown that

$$[\alpha_{0,i+1} + \beta_{0,i+1}(q_i - \Lambda)][\alpha_{2i} + \beta_{2i}(q_{i+1} - \Lambda)]$$

can be made > 0 for all Λ and all i. This is exactly the condition required in the proof of Theorem 4.16. Thus we know that

$$h < (48 / 5(Q + |\Lambda|))^{\frac{1}{2}}$$

will suffice.

This means that there exists a diagonal matrix D such that $D^{-1}(A - \Lambda B)D = T$ is symmetric with

$$\begin{aligned} d_{11} &= 1 \\ d_{ii} &= (\gamma_2\gamma_3\cdots\gamma_i / \delta_2\delta_3\cdots\delta_i)^{\frac{1}{2}} \qquad i=2,3,\ldots,N \\ \gamma_j &= \alpha_{0j} + \beta_{0j}(q_{j-1} - \Lambda) \\ \delta_j &= \alpha_{2j} + \beta_{2j}(q_{j+1} - \Lambda). \end{aligned}$$

The matrix problem we must solve is

$$(A - \Lambda B)Y = 0$$

or $$D^{-1}(A - \Lambda B)DD^{-1}Y = 0.$$

Letting $D^{-1}Y = Z$, the above becomes

$$TZ = 0. \qquad (4.3.18)$$

For any pair (λ, y) of the differential equation we have

$$(A - \lambda B)\bar{y} = \bar{\tau}(\bar{y})$$

or $$D^{-1}(A - \lambda B)DD^{-1}\bar{y} = D^{-1}\,\bar{\tau}(\bar{y}).$$

Multiplying by Z^t, we get

$$Z^tD^{-1}(A - \lambda B)DD^{-1}\bar{y} = Z^tD^{-1}\,\bar{\tau}(\bar{y}). \qquad (4.3.19)$$

Letting $\bar{y} = Y + \Delta y$ and $\lambda = \Lambda + \Delta\lambda$, and substituting into (4.3.19) we get

$$Z^tD^{-1}\{A - \Lambda B - \Delta\lambda B\}D(Z + D^{-1}\Delta y) = Z^tD^{-1}\,\bar{\tau}(\bar{y}).$$

Multiplying out, the expression becomes

$$Z^tD^{-1}(A - \Lambda B)DZ + Z^tD^{-1}(A - \Lambda B)DD^{-1}\Delta y - Z^tD^{-1}\Delta\lambda BDD^{-1}Y$$
$$- Z^tD^{-1}\Delta\lambda BDD^{-1}\Delta y = Z^tD^{-1}\,\bar{\tau}(\bar{y}). \qquad (4.3.20)$$

The first term of (4.3.20) is zero directly from (4.3.18). The second term is zero because

$$Z^tD^{-1}(A - \Lambda B)DD^{-1}\Delta y = Z^tT(D^{-1}\Delta y),$$

but TZ = 0 so

$$(TZ)^t = Z^tT^t = Z^tT = 0.$$

Thus from (4.3.20) comes the identity

$$\Delta\lambda = -\,Z^tD^{-1}\,\bar{\tau}(\bar{y})\;/\;Z^tD^{-1}B\bar{y}. \qquad (4.3.21)$$

We have assumed here that $Z^tD^{-1}B\bar{y} \neq 0$. This assumption is assured if the following two conditions are met.

First, the term $Y^tD^{-1}D^{-1}B\bar{y}$ must not be inherently zero due to orthogonality. Since the eigenvector can be normalized in any fashion, we choose to require that $Y^tD^{-1}D^{-1}BY = 1$. From Theorem 4.19 we now see that

$$\begin{aligned} Y^tD^{-1}D^{-1}B\bar{y} &= Y^tD^{-1}D^{-1}B\{Y + O(h^4)\} \\ &= 1 + O(h^4) \end{aligned}$$

and therefore not equal to zero.

Second, the mesh points $\{x_j\}$ cannot all be at nodes of the eigenfunction y(x) or the term in question would again be zero. This only requires that h be small enough to ensure that when finding the k<u>th</u> eigenvalue there are more than k-1 mesh points. This requirement is necessary in any event because to find the k<u>th</u> eigenvalue with any accuracy our matrix problem must at least be of size k.

If we make the additional assumption that the fourth term of (4.3.20), which contains the product $\Delta\lambda\Delta y$, is negligible compared to the other terms, we get

$$\begin{aligned} \Delta\lambda &\approx - Z^tD^{-1} \; \bar{\tau}(\bar{y}) \;/\; Z^tD^{-1}BY \\ &= - Y^tD^{-1}D^{-1} \; \bar{\tau}(\bar{y}) \;/\; Y^tD^{-1}D^{-1}BY. \qquad (4.3.22) \end{aligned}$$

Since Y is known while $\bar{y}$ is not, (4.3.22) provides an approximation for the error in the matrix eigenvalue which can be computed. This estimate is used in Chapter 5 as part of our adaptive method.

4.3.6 Solution of the Matrix Problem

Again, as with the two uniform methods, we use Rayleigh quotient iteration to solve the matrix problem (4.3.2). Since A and B are no longer symmetric we can expect only linear convergence, according to Ostrowski, with Algorithm 4.5. In order to get cubic convergence with general A and B we must use the generalized Rayleigh quotient in which

$$\Lambda = v^t Au \;/\; v^t Bu$$

The u and v are right and left eigenvectors respectively. Ostrowski goes on, however, to show that if the problem has real eigenvalues with linear elementary divisors then the simple Rayleigh quotient iteration (Alg 4.5) produces quadratic convergence. He argues that since finding both left and right eigenvectors takes approximately twice as much work, we can perform 2 iterations of the simple scheme for every generalized iteration. Thus the simple method gives effectively fourth order convergence when compared with the generalized method.

Since by Theorem 4.16 we know that our problem has real, distinct eigenvalues when h becomes small enough, and therefore has linear elementary divisors, we again use Algorithm 4.5 to solve the matrix problem (4.3.2). Although quadratic convergence is expected, actual numerical tests show that cubic convergence is again obtained during at least some stage of the iteration process.

In Tables 4.3 and 4.4 we present rate of convergence results for the following problem.

$$y'' + (\lambda - x|x|)y = 0$$

$$y(-1) = y(1) = 0$$

Table 4.3 is for $\lambda_0 \approx 2.46258070$ and Table 4.4 for $\lambda_2 \approx 22.2077749$.

In each table λ_i was computed by varying sizes of non-uniform meshes. For each matrix size N, the Rayleigh quotient iteration was performed until

$$|\Lambda_{i,k} - \Lambda_{i,k+1}| < 10^{-13}.$$

β was then computed from the equation

$$|\Lambda_{i,k} - \Lambda_{i,k+1}| = |\Lambda_{i,k-1} - \Lambda_{i,k}|^{\beta}.$$

We see from Tables 4.3 and 4.4 that cubic convergence or better is obtained for each matrix at some point in the iteration. This can be explained by the close association between our matrix problem and the differential equation.

Table 4.3

Convergence Results for λ_0

N = 8		N = 16		N = 20		N = 24	
k	β	k	β	k	β	k	β
2	-	2	20.4	2	2.71	2	1.93
3	15.5	3	2.17	3	3.23	3	3.65
4	15.6			4	2.37	4	2.40
5	∞			5	1.69	5	1.84

N = 28		N = 30		N = 32		N = 34	
k	β	k	β	k	β	k	β
2	2.11	2	2.27	2	2.40	2	2.46
3	3.67	3	3.55	3	3.37	3	3.26
4	2.39	4	2.33	4	2.29	4	2.28
5	1.65	5	1.70	5	1.77	5	1.74

Table 4.4

Convergence Results for λ_2

N = 8		N = 16		N = 32	
k	β	k	β	k	β
2	1.79	2	8.23	2	11.1
3	3.18	3	2.48	3	1.91
4	1.49				

N = 49		N = 61		N = 73	
k	β	k	β	k	β
2	2.62	2	2.28	2	2.27
3	3.29	3	3.70	3	3.51
4	2.30	4	2.29	4	2.26

Our problem can be written in operator notation as

$$L\, y = \lambda\, y \qquad (4.3.23)$$

where L is the Sturm-Liouville operator

$$Lu = \frac{1}{r(x)}[-(p(x)u')' + q(x)u]$$

p, p', q, and r real-valued and continuous, and $p(x) > 0,\ r(x) > 0$ for $a \le x \le b$.

Thus $L: U \to H$ where

$$H = \{u(x) : \int_a^b |u(x)|^2 r(x)\, dx < \infty\}$$ and

$U = \{u(x) : u \in C^2(a \le x \le b)$ which satisfy the boundary conditions$\}$

and where

$$(u,v) = \int_a^b u(x)\ v(x)\ r(x)\ dx.$$

We can now define Rayleigh quotient iteration for (4.3.23).

$$\{L - \lambda^{(k)} r(x)\} z^{(k+1)} = r(x) y^{(k)} \qquad (4.3.24)$$

$$\lambda^{(k+1)} = \int z^{(k+1)} L z^{(k+1)} dx \,/\, (z^{(k+1)}, z^{(k+1)})$$

$$y^{(k+1)} = z^{(k+1)} / \nu$$

where ν is some normalization factor. All integrals are assumed to be over (a,b).

Since $y^{(k)}$ and $z^{(k+1)}$ are in U, and L is the S-L operator, it is well known that each has the expansion

$$y^{(k)}(x) = \sum_{i=1}^{\infty} a_i \, y_i(x) \tag{4.3.25}$$

$$z^{(k+1)}(x) = \sum_{i=1}^{\infty} b_i \, y_i(x) \tag{4.3.26}$$

where the $\{y_i(x)\}$ are the eigenfunctions of L and where $a_i = (y^{(k)}, y_i)$ and $b_i = (z^{(k+1)}, y_i)$. All Σ are assumed to be over 1 to ∞.

Since the normalization factor ν can be chosen arbitrarily, we assume that $y^{(k)}$ has been normalized such that

$$y^{(k)} = y_j + \Sigma' \epsilon_i \, y_i$$

where Σ' denotes $\sum_{\substack{i=1 \\ i\neq j}}^{\infty}$ and $\epsilon_i = \epsilon \cdot d_i = O(\epsilon)$. (i.e. each coefficient is assumed to be a multiple of some small number ϵ.)

We wish to show that after one iteration of this process

$$y^{(k+1)} = y_j + \Sigma' \, O(\epsilon^3) \, y_i .$$

Substituting (4.3.25) and (4.3.26) into (4.3.24) we get

$$\{L - \lambda^{(k)} r\} \; \Sigma b_i y_i = r \; \Sigma a_i y_i .$$

Thus,

$$L[\Sigma b_i y_i] - \lambda^{(k)} r \; \Sigma b_i y_i = r \; \Sigma a_i y_i .$$

Now because the operator L is a bounded, linear operator we can write

$$\Sigma b_i L y_i - \lambda^{(k)} \Sigma b_i r y_i = \Sigma a_i r y_i .$$

But $Ly_i = \lambda_i r y_i$, from which it follows that

$$\Sigma b_i (\lambda_i - \lambda^{(k)}) r y_i = \Sigma a_i r y_i$$

and since the eigenfunction expansion is unique we have

$$b_i = a_i / (\lambda_i - \lambda^{(k)}), \; i=1,2,\ldots$$

Thus

$$\begin{aligned} z^{(k+1)} &= \Sigma b_i y_i \\ &= y_j / (\lambda_j - \lambda^{(k)}) + \Sigma' y_i [\epsilon_i / (\lambda_i - \lambda^{(k)})] \end{aligned}$$

Then when $z^{(k+1)}$ is normalized to yield $y^{(k+1)}$ we find

$$y^{(k+1)} = y_j + (\lambda_j - \lambda^{(k)}) \Sigma' y_i [\epsilon_i / (\lambda_i - \lambda^{(k)})] \tag{4.3.27}$$

Now

$$\lambda^{(k)} = \int y^{(k)} L y^{(k)} dx / (y^{(k)}, y^{(k)}).$$

Substituting the series for $y^{(k)}$ we find that

$$\lambda^{(k)} = \Sigma\Sigma a_i a_j \lambda_j \int y_i r y_j dx / \Sigma\Sigma a_i a_j \int y_i r y_j dx$$

Because of the orthogonality of the eigenfunctions, i.e. $(y_i, y_j) = \delta_{ij}$, we have

$$\begin{aligned} \lambda^{(k)} &= \Sigma a_i^2 \lambda_i / \Sigma a_i^2 \\ &= [\lambda_j + \Sigma' a_i^2 \lambda_i] / [1 + \Sigma' a_i^2] \\ &= [\lambda_j + \epsilon^2 \Sigma' d_i^2 \lambda_i] / [1 + \epsilon^2 \Sigma' d_i^2]. \end{aligned} \tag{4.3.28}$$

Now because the series (4.3.25) is convergent, the series in both numerator and denominator of (4.3.28) converge and thus

$$\lambda^{(k)} = [\lambda_j + O(\epsilon^2)] / [1 + O(\epsilon^2)] \qquad (4.3.29)$$
$$= \lambda_j + O(\epsilon^2).$$

Thus substituting into (4.3.27) we find that

$$y^{(k+1)} = y_j + \Sigma' y_i \, O(\epsilon^3),$$

showing that cubic convergence is obtained.

We note in Tables 4.3 and 4.4 that cubic convergence is indeed obtained. We also note, however, that in every column the convergence rate attains the cubic level and then immediately drops off toward unity.

Suppose that instead of exact solutions $y^{(k)}$ we can only find approximations

$$\bar{y}^{(k)} = y^{(k)} + h^4 e(x) = y^{(k)} + O(h^4)$$

where $e \in U$, i.e.

$$e(x) = \Sigma c_i y_i(x). \qquad (4.3.30)$$

Then

$$\lambda^{(k)} = \frac{\int [y^{(k)} + h^4 e] L [y^{(k)} + h^4 e] dx}{\int [y^{(k)} + h^4 e] r [y^{(k)} + h^4 e] dx}$$
$$= \frac{\int [\Sigma(a_i + h^4 c_i) y_i][\Sigma \lambda_i (a_i + h^4 c_i) r y_i] dx}{\int [\Sigma(a_i + h^4 c_i) y_i][\Sigma(a_i + h^4 c_i) r y_i] dx}$$

$$= \frac{\Sigma\Sigma(a_i + h^4c_i)(a_j + h^4c_j)\lambda_j \int y_i r y_j \, dx}{\Sigma\Sigma(a_i + h^4c_i)(a_j + h^4c_j)\int y_i r y_j \, dx}$$

$$= \frac{\Sigma(a_i + h^4c_i)^2\lambda_i \, dx}{\Sigma(a_i + h^4c_i)^2 \, dx}$$

$$= \frac{\Sigma a_i^2\lambda_i + 2h^4\Sigma a_i c_i \lambda_i + h^8\Sigma c_i^2\lambda_i}{\Sigma a_i^2 + 2h^4\Sigma a_i c_i + h^8\Sigma c_i^2}$$

and thus since (4.3.25) and (4.3.30) converge we have

$$\lambda^{(k)} = \frac{\lambda_j + O(\epsilon^2) + O(h^4)}{1 + O(\epsilon^2) + O(h^4)}$$

$$= \lambda_j + O(\epsilon^2) + O(h^4).$$

Substituting this into (4.3.27) we see that

$$y^{(k+1)} = y_j + \Sigma' y_i[O(\epsilon^2) + O(h^4)]\epsilon_i \,/\, (\lambda_i - \lambda^{(k)})$$

Thus for any fixed h we find that while

$$O(h^4) << O(\epsilon^2)$$

the convergence is cubic as before, however, after several iterations when

$$O(h^4) > O(\epsilon^2)$$

the rate becomes simply linear because $[O(\epsilon^2) + O(h^4)]\epsilon_i$ is $O(\epsilon)$.

This phenomenon is borne out by the results of Tables 4.3 and 4.4.

4.3.7 Deferred Correction

With the use of the estimate (4.3.22) for $\Delta\lambda$ and with an approximation $\bar{T}(Y)$ to $\bar{\tau}(\bar{y})$ we can again apply the deferred correction to improve our final matrix solution.

The algorithm we use is as follows.

ALGORITHM 4.21: Let Y and Λ be the solution to the matrix problem (4.3.2).

1. Estimate $\bar{\tau}(\bar{y})$ by computing $\bar{T}(Y)$
2. Estimate $\Delta\lambda$ by computing

$$\overline{\Delta\lambda} = - Z^tD^{-1}\bar{T}(Y) \;/\; Z^tD^{-1}BY$$

3. Let $\bar{\lambda} = \Lambda + \overline{\Delta\lambda}$
4. Solve $(A - \bar{\lambda}B)\bar{\bar{y}} = \bar{T}(Y)$ for $\bar{\bar{y}}$

If $\Delta\lambda$ were found exactly in Step 2 using exprssion (4.3.21) rather than (4.3.22) then we could compute $\lambda = \Lambda + \Delta\lambda$ and $\bar{y}$ exactly. However, we must use instead the expression

$$\overline{\Delta\lambda} = -Y^tD^{-1}D^{-1}\bar{T}(Y) \;/\; Y^tD^{-1}D^{-1}BY$$

The errors in this expression occur in two places. First, concerning the second Y in the denominator, we know from Theorem 4.19 that $Y = \bar{y} + O(h^4)$. Second, there is an error when $\bar{T}(Y)$ is used to approximate $\bar{\tau}(\bar{y})$.

We have not yet introduced an approximation $\bar{T}(Y)$ to $\bar{\tau}(\bar{y})$, choosing rather to present it along with the practical details of the adaptive method of Chapter 5. We assume for now that such an approximation is available and that

$$\bar{T}(Y) = \bar{\tau}(\bar{y}) + O(h^8)$$

recalling that $\bar{\tau}(\bar{y}) = O(h^5)$.

With these two sources of error

$$\overline{\Delta\lambda} = -Y^tD^{-1}D^{-1}[\ \bar{\tau}(\bar{y}) + O(h^8)]/Y^tD^{-1}D^{-1}B[\bar{y} + O(h^4)]$$

At this point we note that as $h \to 0$ the number of mesh points $N \to \infty$ as $1/h$. Thus $Y^tD^{-1}D^{-1}BY = O(h)$ and $Y^tD^{-1}D^{-1}[O(h^8)] = O(h^7)$ and so

$$\overline{\Delta\lambda} = \frac{-Y^tD^{-1}D^{-1}\ \bar{\tau}(\bar{y})}{Y^tD^{-1}D^{-1}B\bar{y} + O(h^5)} + \frac{O(h^7)}{O(h)}$$

$$= \Delta\lambda\ \{1 + O(h^4)\} + O(h^6)$$

$$= \Delta\lambda + O(h^6) \qquad \text{since } \Delta\lambda = O(h^4).$$

$$\therefore\ \Lambda + \overline{\Delta\lambda} = \Lambda + \Delta\lambda + O(h^6)$$

$$= \lambda + O(h^6)$$

Thus we have shown that while $\Lambda = \lambda + O(h^4)$ one correction step gives us sixth order accuracy as before.

The non-uniform finite-difference method developed in this section becomes the basis of our adaptive Sturm-Liouville solver of Section 5.2. Numerical results from the method are presented after the adaptive algorithm is discussed.

The contributions of this chapter have been to provide us with i) a finite-difference method which uses the non-uniform mesh we expect from an adaptive method, ii) a cubically convergent technique for solving the resulting matrix problem, and iii) a process for correcting the matrix solution which increases the accuracy from $O(h^4)$ to $O(h^6)$.

Chapter 5

AUTOMATIC METHODS

5.1 Introduction

Davis and Rabinowitz[12], in their book, define four categories of numerical integration methods. These definitions also apply to finite-difference methods for solving the Sturm-Liouville problem. They are:

Adaptive - A finite-difference solution method is adaptive if the points of the mesh are chosen to depend on the nature of the solution, (y,λ).

Non-adaptive - A method is non-adaptive if the mesh is chosen according to a fixed scheme independent of the solution.

Iterative - A method is iterative if successive approximations to the solution are made until agreement within a prescribed tolerance is achieved.

Non-iterative - A method is non-iterative if information taken from the first approximation is used to generate a second, which is then taken as the final result.

With these definitions we are able to categorize the methods of Chapter 4 as non-adaptive, non-iterative methods. The mesh is chosen before using the algorithms and then the deferred correction is performed once to get a final result.

In this chapter we present two iterative methods, a non-adaptive, iterative method in Section 5.2 and an adaptive, iterative method in Section 5.3.

Both methods require accurate initial estimates to ensure convergence, and neither method can be used to solve singular S-L problems. Such a method we define as non-automatic. We say that a method is automatic if, when supplied with only the differential equation, boundary conditions, and a desired tolerance level, it can return an approximate solution, (Y, Λ), accurate to within the specified tolerance on λ.

In Chapters 6 and 7 we address these shortcomings. The final result is an automatic, adaptive finite-difference method for solving the S-L problem.

5.2 A Non-adaptive, Iterative Method

In this section we present a non-automatic, non-adaptive, iterative method.

5.2.1 Background

The algorithm to be introduced is motivated by the work of Lentini and Pereyra[26,27]. In these papers, a variable order finite-difference method is developed for solving nonlinear multipoint boundary value problems. Eigenvalue problems are not considered.

The general problem solved in that work is the nonlinear, first order system

$$y'(t) - f(t, y(t)) = 0, \quad a \le t \le b$$

subject to the multipoint boundary conditions

$$g(y(t_1),y(t_2),\ldots,y(t_N)) = 0,\ a \leq t_1 <\ldots< t_N \leq b.$$

The method is based on the technique of deferred corrections. It is assumed that the following asymptotic expansion for the local truncation error is available.

$$\tau_h(x_i) = -\sum_{k=1}^{K} a_k y^{(2k+2)}(x_i)h^{2k} + O(h^{2K+2}) \tag{5.2.1}$$

The first m terms of (5.2.1) can then be approximated by a linear combination of function values at various points as such:

$$\begin{aligned} T_m(x_i) &= -\sum_{k=1}^{m} a_k y^{(2k+2)}(x_i)h^{2k+2} \\ &= \sum_{s=1}^{2m+2+q} w_s y\,(x_i + \alpha_s h) + O(h^q) \qquad (5.2.2) \\ &= S_m(y\,(x_i)) + O(h^q) \end{aligned}$$

where the α_s are integers. The numerical approximation is found by discretizing the differential equation and solving the resulting system.

If $F_h(Y) = 0$ is defined as the nonlinear discretized problem and $Y = (Y_1,Y_2,\ldots,Y_N)^t$ as the discretized solution, we find that Pereyra in [32] introduces the following iterated deferred correction algorithm.

ALGORITHM 5.1:

1. Let $Y^{(k)}$ be an $O(h^{2k+2})$ discrete solution.
2. Compute $h^{-2}S_{k+1}(Y^{(k)})$, an $O(h^{2k+2})$ approximation

to the first k+1 terms of the local truncation error.

3. Solve $F_h(Y) = h^{-2}S_{k+1}(Y^{(k)})$ for $Y^{(k+1)}$.

It is obvious that only a limited number of iterations of Algorithm 5.1 can be performed on a mesh of size N, because an appropriate number of mesh points must be available surrounding each point to form $S_{k+1}(Y^{(k)})$. Pereyra also notes that for a given problem and mesh size, only a limited number of corrections will produce improvements to the solution. As Pereyra puts it, this is due to, "the growth of high order differences".

Equipped then with an estimate Δ_{k-1} to the error $e_{k-1} = Y^{(k-1)} - \bar{y}$, Lentini and Pereyra introduce the following iterative algorithm.

```
ALGORITHM 5.2:
  1. Let $k = 0$.
  2. Is $N \geq 4$ ?
       NO: Refine mesh and GO TO 1.
       YES: Then $S_1(Y^{(0)})$ can be computed, so solve
            $F_h(Y) = 0$ for $Y^{(0)}$, compute $S_1(Y^{(0)})$,
            and compute $\Delta_0$.
  3. If $\| \Delta_0 \| <$ TOL (the user provided tolerance)
     then STOP.

  Correction Loop:
  4. Let $k = k+1$.
```

5. If $N < 2k+2$ then refine the mesh and GO TO 1. (not enough points to compute S_k)
6. Solve $F_h(Y) = S_k(Y^{(k-1)})$ for $Y^{(k)}$.
7. Compute $S_{k+1}(Y^{(k)})$.
8. Compute Δ_k.
9. If $\| \Delta_k \| <$ TOL then STOP.
10. If $\| \Delta_k \| \leq C \| \Delta_{k-1} \|$, $(0 < C \leq 1)$, then GO TO 4 else refine mesh and GO TO 1. (This ensures that the errors are decreasing, i.e. it checks for growth of high order differences)

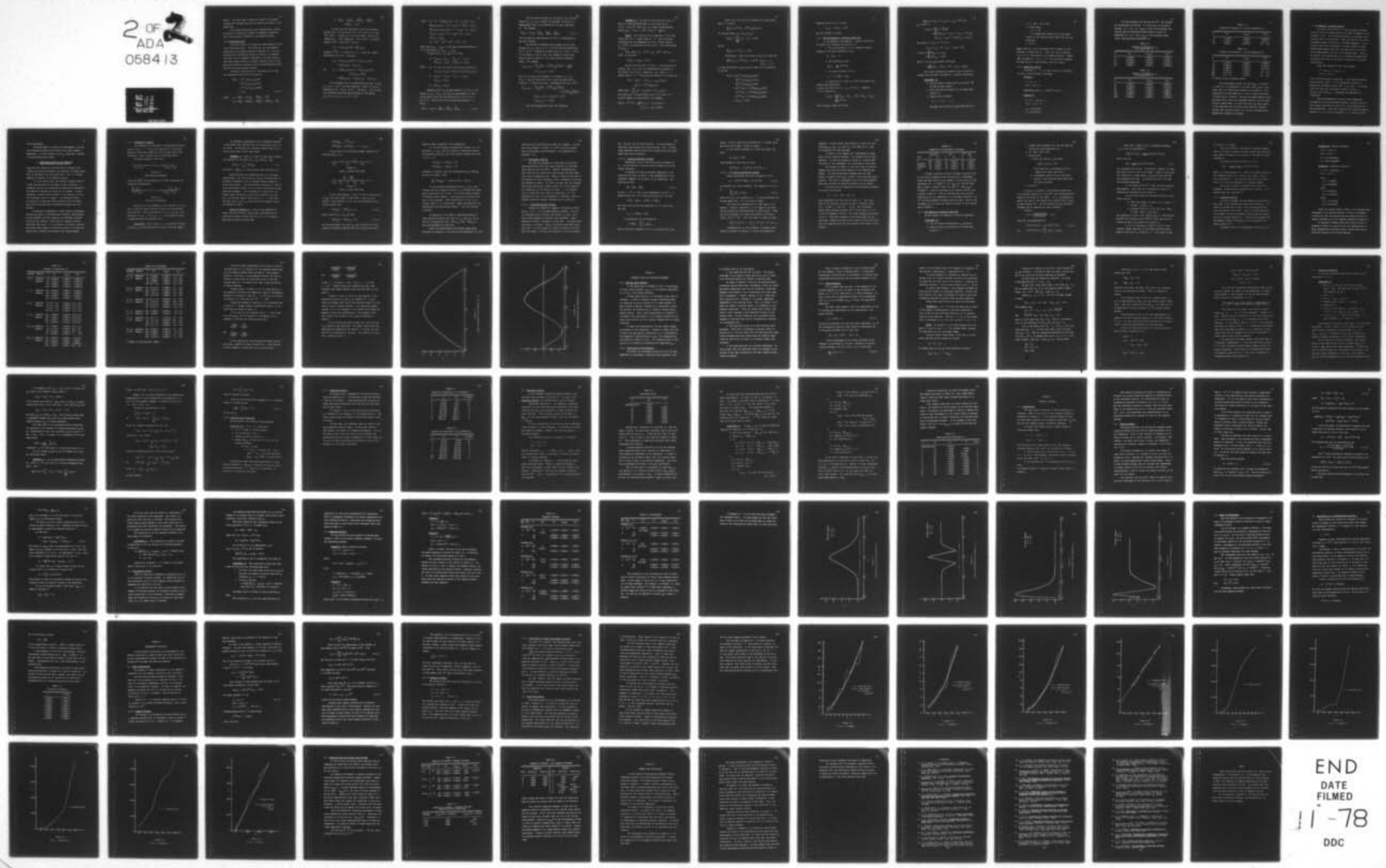
Pereyra admits that there are some theoretical difficulties in obtaining the expansions necessary to justify the method. Different differentiation formulas must be used at different points. This problem is at least delayed by replacing higher derivatives of y by derivatives of $f(x,y)$ two orders lower.

A further difficulty we encounter when trying to use Algorithm 5.2 to solve our problem is that Pereyra only considers uniform spacing when computing the coefficients w_s of (5.2.2). While this is sufficient for the present non-adaptive algorithm, it is not for our ultimate adaptive method. Lentini and Pereyra in [26] and [27] do consider a non-uniform mesh, however, there again only first order systems are discussed.

We should also recall that our objective is not, as was Lentini and Pereyra's, to develop a variable order

AD-A058 413 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO F/G 12/1
AN ADAPTIVE FINITE-DIFFERENCE METHOD FOR SOLVING THE STURM-LIOU--ETC(U)
MAY 78 D A NELSON
UNCLASSIFIED AFIT-CI-78-22 NL

2 OF 2
ADA
058413



END
DATE
FILMED
11-78
DDC

method. We rather want to keep the order of the method constant and increase accuracy by refining the mesh in the proper way.

With these difficulties and considerations in mind, we choose to introduce a simpler non-adaptive algorithm which can be easily converted to an adaptive method.

5.2.2 Truncation Error

Our first task is to develop an approximation to the local truncation error. The basis for our non-adaptive algorithm is the Numerov method of Section 4.2. The non-uniform finite-difference method of Section 4.3 becomes the basis for our adaptive algorithm of the next section. Since the Numerov method is simply a special case of this non-uniform method we develop here the approximation necessary for the more general problem.

We recall from Section 4.3 (equation 4.3.8) that the truncation error can be written as

$$\begin{aligned}\tau(x_i) &= y^{(5)}(x_i)h_i h_{i+1} a_i/180 \\ &+ y^{(6)}(x_i)h_i h_{i+1} b_i/720 \\ &+ y^{(7)}(x_i)h_i h_{i+1} c_i/5040 \\ &+ O(h^8) \qquad (5.2.3)\end{aligned}$$

where
$$a_i = 2h_{i+1}^3 + 3h_i h_{i+1}^2 - 3h_i^2 h_{i+1} - 2h_i^3$$
$$b_i = 3h_{i+1}^4 + 2h_i h_{i+1}^3 - 7h_i^2 h_{i+1}^2 + 2h_i^3 h_{i+1} + 3h_i^4$$

$$c_i = 5h_{i+1}^5 + 2h_ih_{i+1}^4 - 9h_i^2h_{i+1}^3 + 9h_i^3h_{i+1}^2$$
$$-2h_i^4h_{i+1} - 5h_i^5.$$

We know from the derivation of the non-uniform method that it is exact for $y(x) = x^p$, $p=0,1,2,3,4$, and thus $\tau(x_i) = 0$ when $y(x)$ is of this form. Now let us compute $t_{ip} = \tau(x_i)$ when $y(x) = x^p$, $p=5,6,7$. (or equivalently $y''(x) = p(p-1)x^{p-2}$, $p=5,6,7$)

$$t_{i5} = 5!\,h_ih_{i+1}a_i/180 = \tfrac{2}{3}h_ih_{i+1}a_i$$

because $y^{(5)}(x) = 5!$ when $y(x) = x^5$, while all higher derivatives are zero. Similarly,

$$t_{i6} = 6!\,x_ih_ih_{i+1}a_i/180 + 6!\,h_ih_{i+1}b_i/720$$
$$= 4x_ih_ih_{i+1}a_i + h_ih_{i+1}b_i$$

and
$$t_{i7} = 7!\,x_i^2h_ih_{i+1}a_i/360 + 7!\,x_ih_ih_{i+1}b_i/720$$
$$+7!\,h_ih_{i+1}c_i/5040$$
$$= 14x_i^2h_ih_{i+1}a_i + 7x_ih_ih_{i+1}b_i + h_ih_{i+1}c_i$$

We now assume that the approximate solution, Λ and Y_i, $i=0,1,\ldots,N+1$, has been computed. From it Y_i'' can be computed by $Y_i'' = (q(x_i) - \Lambda)Y_i$. Letting $f_i = Y_i''$, we form the following interpolating polynomial on the points $\{x_j,\ j=i-3,\ i-2,\ i-1,\ i,\ i+1,\ i+2\}$:

$$g_i(x) = f_i + (x - x_i)f[x_{i-1},x_i] + (x - x_{i-1})(x - x_i)\cdot$$
$$f[x_{i-1},x_i,x_{i+1}] + (x - x_{i-1})(x - x_i)(x - x_{i+1})\cdot$$
$$f[x_{i-2},x_{i-1},x_i,x_{i+1}] + (x - x_{i-2})\ldots(x - x_{i+1})\cdot$$
$$f[x_{i-2},\ldots x_{i+2}] + (x - x_{i-2})\ldots(x - x_{i+2})\cdot$$
$$f[x_{i-3},\ldots x_{i+2}]$$
$$= A_i + B_i x + C_i x^2 + D_i x^3 + E_i x^4 + F_i x^5$$

where $f[x_j,x_{j+1},\ldots x_{j+k}]$ is the <u>kth</u> divided difference of Y'' at the points x_j, $x_{j+1},\ldots,$ x_{j+k}.

Solving for D_i, E_i, and F_i, we find

$$D_i = f[x_{i-2},\ldots x_{i+1}] - (x_{i-2} + \ldots + x_{i+1})\cdot$$
$$f[x_{i-2},\ldots x_{i+2}] + s_i f[x_{i-3},\ldots x_{i+2}]$$

where $$s_i = (x_{i-2}x_{i-1}+x_{i-2}x_i+x_{i-2}x_{i+1}+x_{i-2}x_{i+2}+x_{i-1}x_i+$$
$$x_{i-1}x_{i+1}+x_{i-1}x_{i+2}+x_ix_{i+1}+x_ix_{i+2}+x_{i+1}x_{i+2})$$

$$E_i = f[x_{i-2},\ldots x_{i+2}] - (x_{i-2} + \ldots + x_{i+2})\cdot$$
$$f[x_{i-3},\ldots x_{i+2}]$$

$$F_i = f[x_{i-3},\ldots x_{i+2}]$$

Regarding $g_i(x)$ as an approximation to $y''(x)$ on the points $\{x_{i-3},\ldots,x_{i+2}\}$, we can now approximate the truncation error $\tau(x_i)$ by substituting $g_i(x)$ into (5.2.3) for $y''(x)$. This yields the following approximation to $\tau(x_i)$.

$$\tau(x_i) \approx T_i(x_i) = \frac{D_i}{20}t_{i5} + \frac{E_i}{30}t_{i6} + \frac{F_i}{42}t_{i7} \qquad (5.2.4)$$

For the several points at the ends of the interval where $f[x_{i-3},\dots x_{i+2}]$ cannot be evaluated, we choose to approximate $\tau(x_i)$ by extrapolating the last computable T_i. For example,

$$\tau(x_2) \approx T_3(x_2) = \frac{D_3}{20}t_{25} + \frac{E_3}{30}t_{26} + \frac{F_3}{42}t_{27} \qquad (5.2.5)$$

Just how good an approximation we have is established in the next theorem.

The divided differences which appear in the coefficients D_i, E_i, and F_i of (5.2.4) are divided differences of Y'', not y''. Theorem 4.19 shows that $Y''_i = y''(x_i) + O(h^4)$. If the errors in the Y_i were just random, we would find that we lose an order of h for each divided difference taken. For example,

$$f[x_{i-1},x_i] = \frac{Y''_i - Y''_{i-1}}{h_i} = \frac{y''(x_i) - y''(x_{i-1})}{h_i} + \frac{O(h^4)}{h_i}$$

$$= y''[x_i,x_{i-1}] + O(h^3).$$

But, if the errors are assumed to be smooth, i.e. $Y''_i = y''(x_i) + e(x_i)h^4$ where $e(x)$ has at least as many derivatives as y'', then all divided differences are still $O(h^4)$.

$$f[x_{i-1},x_i] = \frac{Y''_i - Y''_{i-1}}{h_i} = \frac{y''(x_i) - y''(x_{i-1})}{h_i} + h^4\,\frac{e(x_i) - e(x_{i-1})}{h_i}$$

$$= y''[x_{i-1},x_i] + e'(x_i)h^4 + O(h^5)$$

$$= y''[x_{i-1},x_i] + O(h^4).$$

With this assumption we get the following:

<u>THEOREM 5.3</u>: Let $y(x)$ be the solution to (4.1.1) and be 8 times differentiable on $[a,b]$ and let $Y_i'' = y''(x_i) + e(x_i)h^4$, where $e(x)$ is 6 times differentiable. Then $T_i(x_i) - \tau(x_i) = O(h^8)$, where $h = \max_i(h_i)$.

<u>Proof</u>: We let $\overline{T}_i(x_i)$ be the quantity (5.2.4) computed with $y''[\ldots]$ rather than $f[\ldots]$. From the above discussion and the assumption on $e(x)$, we know that $\overline{D}_i = D_i + O(h^4)$, and similarly for $\overline{E}_i$ and $\overline{F}_i$. (the coefficients of $\overline{T}_i(x_i)$)

Thus, since $t_{i5} = O(h^5)$, $t_{i6} = O(h^6)$, and $t_{i7} = O(h^7)$, we see that

$$\overline{T}_i(x_i) = T_i(x_i) + O(h^9). \qquad (5.2.6)$$

We now find the error in $\overline{T}_i(x_i)$. Our procedure for finding $\tau(x_i)$ is to fit an interpolating polynomial to the points $\{(x_i, y''(x_i)\}$, namely $\overline{g}_i$. (g_i, using $y''[\ldots]$ rather than $f[\ldots]$) From Conte and deBoor[10] we know that

$$y''(x) = \overline{g}_i(x) + y''[x_{i-3},\ldots x_{i+2},x]P_6(x)$$
$$= \overline{g}_i(x) + y^{(8)}(\theta)P_6(x)/6!$$

where $P_6(x) = \prod_{j=i-3}^{i+2}(x - x_j)$ and $\theta \in [x_{i-3},x_{i+2}]$.

Thus when $\overline{g}_i(x)$ is substituted into (5.2.3) for $y''(x)$, we must compute its derivatives, for example

$$\overline{g}_i'(x) = y^{(3)}(x) - (\tfrac{d}{dx}y''[x_{i-3},\ldots x_{i+2},x]P_6(x) + y''[x_{i-3},\ldots x_{i+2},x]P_6'(x).$$

Since $T_i(x)$ will only be evaluated at a mesh point, $P_6(x_i) = 0$ and so

$$\bar{g}_i'(x) = y^{(3)}(x) - y^{(8)}(\theta_1)P_6'(x)/8!.$$

At any mesh point, $x_j \in [x_{i-3}, x_{i+2}]$

$$P_6'(x_j) = \prod_{\substack{k=i-3 \\ k\neq j}}^{i+2} (x_j - x_k) = O(h^5)$$

and so

$$\bar{g}_i'(x_j) = y^{(3)}(x_j) + O(h^5).$$

Continuing to take derivatives of $\bar{g}_i(x)$ we find that

$$\bar{g}_i^{(k)}(x_j) = y^{(k+2)}(x_j) + O(h^{6-k}), \quad \begin{array}{l} k=1,2,\dots,5 \\ j=i-3,\dots,i+2 \end{array}$$

i.e.each derivative of $\bar{g}_i$ loses an order of h in accuracy. So we have

$$\begin{aligned}\bar{T}_i(x_i) &= \bar{g}_i^{(3)}(x_i)h_ih_{i+1}a_i/180 + \\ &\quad \bar{g}_i^{(4)}(x_i)h_ih_{i+1}b_i/720 + \\ &\quad \bar{g}_i^{(5)}(x_i)h_ih_{i+1}c_i/5040 \\ &= [y^{(5)}(x_i) + O(h^3)]h_ih_{i+1}a_i/180 + \\ &\quad [y^{(6)}(x_i) + O(h^2)]h_ih_{i+1}b_i/720 + \\ &\quad [y^{(7)}(x_i) + O(h)]h_ih_{i+1}c_i/5040 \\ &= \tau(x_i) + O(h^8).\end{aligned}$$

Combining with (5.2.6) we have

$$T_i(x_i) = \tau(x_i) + O(h^8)$$

and the theorem is proved.

5.2.3 The Non-adaptive, Iterative Algorithm

We now present a non-adaptive, iterative algorithm. We recall the following from Section 4.2.

1. The discretization (4.2.1) (Numerov method) leading to the matrix problem (4.2.2):

$$AY - \Lambda BY = 0.$$

2. The truncation error:

$$\tau(x_i) = \frac{1}{240} h^6 y^{(6)}(\theta).$$

3. The error estimate (4.2.6):

$$\Delta\lambda \approx - Y^t \ \bar{\tau}(\bar{y}) \ / \ Y^t BY.$$

4. Algorithm 4.6, used to correct the matrix eigenvalue and eigenvector.

We also note that when $h_i = h_{i+1}$ for all i , equation (5.2.5) reduces to

$$\begin{aligned} T_i(x_i) &= \frac{h^2}{240} (Y''_{i-2} - 4Y''_{i-1} + 6Y''_i - 4Y''_{i+1} + Y''_{i+2}) \\ &= \frac{h^6}{240} y^{(6)}(\theta). \end{aligned}$$

This is easily shown as follows.

When $h_i = h_{i+1} = h$, $t_{i5} = 0$, $t_{i6} = 3h^6$, and $t_{i7} = 21h^6$ so

$$T_i(x_i) = \frac{E_i}{10} h^6 + \frac{F_i}{2} x_i h^6$$
$$= \frac{h^6}{10} \{ f[x_{i-2}, \ldots x_{i+2}] - (x_{i-2} + \ldots + x_{i+2}) \cdot f[x_{i-3}, \ldots x_{i+2}] + 5x_i f[x_{i-3}, \ldots x_{i+2}] \}$$

but since $h_i = h_{i+1} = h$ for all i,

$$(x_{i-2} + x_{i-1} + x_i + x_{i+1} + x_{i+2}) = 5x_i .$$

$$\therefore T_i(x_i) = \frac{h^6}{10} f[x_{i-2}, \ldots x_{i+2}]$$
$$= \frac{h^6}{10} \{\Delta^i f_{i-2} / 4! h^4\}$$

where Δ^i is the i<u>th</u> forward difference

$$= \frac{h^2}{240} \{Y''_{i-2} - 4Y''_{i-1} + 6Y''_i - 4Y''_{i+1} + Y''_{i+2}\}.$$

With these preliminary remarks we are now able to present the following non-adaptive, iterative algorithm.

<u>ALGORITHM 5.4</u>:

1. Set up a basic uniform mesh π_0 on $[a,b]$. (We use N=8 in most cases.)
2. Solve the matrix problem (4.2.2) using Algorithm 4.5.
3. Compute error estimate

$$\Delta\lambda = - Y^t \bar{T}(Y) / Y^t BY$$

and make one correction using Algorithm 4.6.

4. If $|\Delta\lambda| <$ TOL, STOP.

5. Refine mesh:

   i) $N = 2N$

   ii) Interpolate values of Y_i at new mesh points to get starting right-hand side for next iteration.

6. GO TO 2.

<u>Note</u>: Step 5ii) is not necessary but it speeds up convergence of Algorithm 4.5. After the first iteration (i.e. $N > 8$), Algorithm 4.5 is changed slightly. Rather than letting $BY_0 = (1,1,\dots,1)^t$, the algorithm computes BY from vector Y, calculated at step 5ii) above.

5.2.4 Numerical Results

In Tables 5.1 - 5.4 we show results of Algorithm 5.4 run on the following 2 problems.

<u>Problem I</u>:

$$y'' + \lambda y = 0$$
$$y(0) = y(1) = 0$$

Eigenvalues are $\lambda_n = (n+1)^2\pi^2$, $n=0,1,\dots$

<u>Problem II</u>:

$$y'' + (\lambda - x|x|)y = 0$$
$$y(-1) = y(1) = 0$$

$$\lambda_0 \approx 2.46258070$$
$$\lambda_2 \approx 22.2077749$$

For each problem, TOL was set at 10^{-6}. The columns are interpreted as follows. N is the size of the matrix problem, $|e| = |\Lambda - \lambda|$, $|e_{est}|$ is the error estimate calculated from the expression shown above in Step 3 of Algorithm 5.4, $|e_c| = \Lambda + e_{est}$ is the corrected eigenvalue resulting from Algorithm 4.6.

Table 5.1

Results for Problem I, λ_0
With Algorithm 5.4

N	$\|e\|$	$\|e_{est}\|$	$\|e_c\|$
8	9.839D-4	9.592D-4	2.471D-5
16	6.122D-5	6.091D-5	3.127D-7
32	3.825D-6	3.817D-6	4.573D-9
64	2.388D-7	2.387D-7	6.899D-11

Table 5.2

Results for Problem I, λ_2
With Algorithm 5.4

N	$\|e\|$	$\|e_{est}\|$	$\|e_c\|$
8	7.479D-1	1.011D0	2.633D-1
16	4.516D-2	4.313D-2	2.031D-3
32	2.794D-3	2.763D-3	3.166D-5
64	1.742D-4	1.737D-3	4.698D-7
128	1.088D-5	1.087D-5	7.266D-9
256	6.801D-7	6.798D-7	2.932D-10

Table 5.3

Results for Problem II, λ_0 With Algorithm 5.4

N	$\|e\|$	$\|e_{est}\|$	$\|e_c\|$
8	2.278D-4	2.232D-4	4.597D-6
16	1.432D-5	1.383D-5	4.85 D-7
32	9.07D-7	8.629D-7	*4. D-8

Table 5.4

Results for Problem II, λ_2 With Algorithm 5.4

N	$\|e\|$	$\|e_{est}\|$	$\|e_c\|$
8	1.871D-1	2.524D-1	6.529D-2
16	1.130D-2	1.079D-2	5.064D-4
32	6.991D-4	6.911D-4	8.02 D-6
64	4.363D-5	4.346D-5	1.7 D-7
128	2.77 D-6	2.720D-6	*5. D-8
256	2.2 D-7	1.701D-7	*5. D-8

* Correct to all available digits

The discrepancy between $|e|$ and $|e_{est}|$, in Tables 5.3 and 5.4, is attributed to the fact that λ_0 and λ_2 are known only to 9 significant figures for Problem II. The tables show us that Algorithm 5.4 can be used effectively to solve the S-I problem. The error estimate appears quite good and serves well as a stopping criterion for the algorithm in these cases. We also note that the final result of the algorithm will be much better than TOL required because the corrected value, $|e_c|$, gives us several more significant figures of accuracy.

5.3 An Adaptive, Iterative Method

We recall from Section 5.1 that an adaptive method is one in which the mesh points are chosen to depend on the solution, (y, λ). This obviously implies, in general, that the mesh will be non-uniform, Thus the need for the finite-difference method of Section 4.3 becomes apparent. If the solution were known beforehand, we could choose the mesh points appropriately and then use Algorithm 4.5 to solve for an approximate solution on our optimal mesh. This is, of course, not possible and so an iterative strategy must be devised which alters the mesh to conform to a converging solution.

Denny and Landis[14] solve the problem

$$u'' + P(u,y)u' + Q(u,y) = 0$$
$$u(0) = 0, \ u(1) = 1$$

with finite-difference techniques. They obtain successive distributions, $\{y_i^{(k)}\}$, of the independent variable by making the truncation error, $T_i^{(k)}$, go to zero. To accomplish this they must continually compute $\partial T_i/\partial y_{i-1}$, $\partial T_i/\partial y_i$, and $\partial T_i/\partial y_{i+1}$. Eigenvalue problems are not considered.

Gary and Helgason[19], who do solve eigenvalue problems with finite-difference methods, require that the user supply functions $s(x)$ and $s'(x)$, which describe the non-uniform mesh. They then transform the resulting matrix problem, $A + \lambda B$, into the standard $A + \lambda I$ and solve with the QR algorithm.

Returning again to Lentini and Pereyra[26], we find the following strategy for solving first order systems adaptively. In each interval (x_i, x_{i+1}) they add J uniformly distributed points, where

$$J = \frac{\text{(truncation error in the interval)}}{\text{(that interval's alloted error)}}$$

They note that unless the alloted error is chosen judiciously and altered throughout the process, too many points will be introduced into the mesh early. This, of course, defeats the purpose of an adaptive method.

It is our belief that the optimal strategy would be to add only one point to the mesh at each iteration. A technique like the one described by Malcolm and Simpson[28] for numerical integration would then be needed. At each iteration, a point would be added to the interval in which the truncation error is largest. An elaborate system of sorting or a heap type data structure would be necessary, and the method would undoubtedly become very inefficient.

We present a compromise to these various techniques. At most one point is introduced per interval, specifically at the midpoint, but all intervals not meeting a certain criterion are refined. Points are not removed from the mesh once they enter. It is believed, by Lentini, Pereyra, and others that removal of points can prove to be quite unstable and in addition can produce very coarse meshes.

5.3.1 Refinement Strategy

Our strategy for refinement is described as follows. Let $x_j < x_{j+1} < \dots < x_{j+k}$ be a sequence of consecutive points in the current mesh which do not meet some given criterion. (Such criteria will be discussed below.)

The mesh currently looks like:

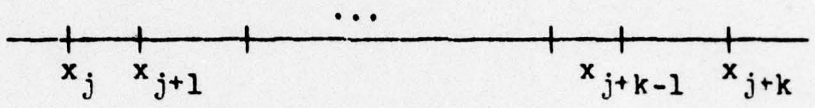


Figure 5.1

Mesh Before Refinement

Points $\{\bar{x}_i\}_{i=j}^{j+k+1}$ are now added to the mesh producing the following configuration.

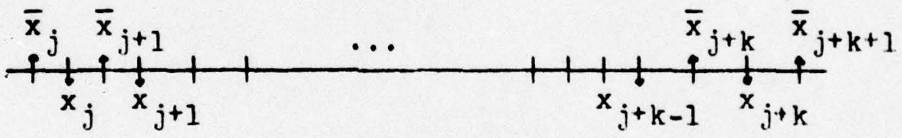


Figure 5.2

Mesh After Refinement

Each $\bar{x}_i$ is at the midpoint of the interval (x_{i-1}, x_i). This is done for each interval of points not satisfying the criterion. The only exception is if x_j or x_{j+k} is an endpoint, i.e. $x_j = a$ or $x_{j+k} = b$. In this case no point is added outside the interval $[a, b]$.

DEFINITION: When a point, x_j, is refined by adding $\bar{x}_j$ and $\bar{x}_{j+1}$ as described above, we say it has been split.

An important consequence of our refinement strategy is the effect that the splitting of a point has on the vector $\bar{\tau}(\bar{y})$. We see from the following theorem that the desired result does in fact take place.

THEOREM 5.5: When x_j is split by the above strategy and replaced by the points $\bar{x}_{j1}$, $\bar{x}_{j2}$, and $\bar{x}_{j3}$, then

$$|\tau(x_j)| > \sum_{k=1}^{3} |\tau(\bar{x}_{jk})| \tag{5.3.1}$$

provided $h = \max_i(h_i)$ is sufficiently small and $\tau(x_j) \neq 0$.

Before proving the theorem we make the following observations. The restriction on $\tau(x_j)$ not being zero is certainly reasonable, for no strategy should attempt to split such a point. The restriction we place on h, that of being sufficiently small, is not a real restriction in most practical problems. It is necessary to insure that the higher order derivatives at the two new points, $\bar{x}_{j1}$ and $\bar{x}_{j3}$, are not much larger than those at x_j. If such were the case then h_j and h_{j+1} would need to be smaller before the theorem is valid.

Proof of Theorem 5.5: Since we are assuming all along that the solution y has as many continuous derivatives as we need, we know that for all $\epsilon > 0$, h can be made small enough to ensure that

$$|y^{(m)}(\bar{x}_{j1}) - y^{(m)}(x_j)| < \epsilon$$

$$|y^{(m)}(\bar{x}_{j3}) - y^{(m)}(x_j)| < \epsilon, \quad m=5,6,\ldots$$

We now look at the physical changes brought on by the splitting of x_j.

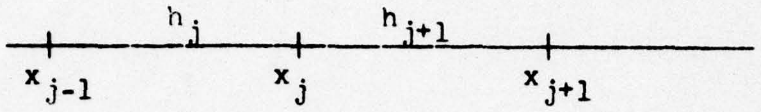


Figure 5.3

Point x_j Before Splitting

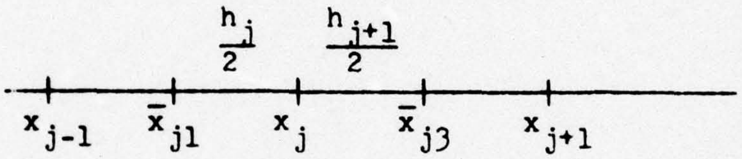


Figure 5.4

Point x_j After Splitting

We see from Figures 5.3 and 5.4 that at the point x_j, or $\bar{x}_{j2}$, the only change is that both h_j and h_{j+1} are halved. From equation (5.2.3) for the truncation error, we can now say that

$$|\tau(\bar{x}_{j2})| < |\tau(x_j)| \,/\, 32 \qquad (5.3.2)$$

and in fact if $h_j = h_{j+1}$ we have

$$|\tau(\bar{x}_{j2})| < |\tau(x_j)| \,/\, 64. \qquad (5.3.3)$$

The two new points, $\bar{x}_{j1}$ and $\bar{x}_{j3}$, are located at the midpoints of existing intervals and thus contain only even terms in their truncation error expansions.

If, in the original configuration (Figure 5.3), we had had $h_j = h_{j+1}$, then the theorem is immediate because the points $\bar{x}_{j1}$ and $\bar{x}_{j3}$ will also satisfy inequalities like (5.3.3), i.e.

$$|\tau(\bar{x}_{j1})| < |\tau(x_j)| / 64$$

$$|\tau(\bar{x}_{j3})| < |\tau(x_j)| / 64$$

provided, of course, that the derivatives at $\bar{x}_{j1}$ and $\bar{x}_{j3}$ are not too large. Thus

$$\sum_{k=1}^{3} |\tau(\bar{x}_{jk})| < 3|\tau(x_j)|/64 < |\tau(x_j|.$$

If the original configuration had $h_i \neq h_{j+1}$ then we know that the truncation error at x_j contains odd terms and in particular an h^5 term. The two new points contain h^6 terms as their largest. They also have h_j's which are half of the original. Thus coupled with (5.3.2) it is obvious that (5.3.1) holds again. (Again provided that the derivatives are sufficiently smooth.) Thus the theorem is proved.

In addition to the smaller truncation errors at these points we should note that at x_{j-1} and x_{j+1} one of the h's is halved and thus $|\tau(x_{j-1})|$ and $|\tau(x_{j+1})|$ are also decreased by splitting.

While the actual amount of decrease seems quite difficult to compute, it is obvious from expression (4.3.21) that the act of splitting will make $|\Delta\lambda|$ smaller. We also know from Theorem 4.19 that $\Delta\lambda = O(h^4)$ and thus several acts of splitting, which decrease h, are guaranteed to decrease $|\Delta\lambda|$.

5.3.2 Refinement Criteria

Now that the mechanics of splitting a point have been described we must have some criterion for deciding which points should be split. Several strategies have been tried and we present each here along with some numerical results comparing them. Based on these results and on the analysis of each strategy a "best" strategy for solving our problem is chosen. By "best" we mean here the strategy which most efficiently (fewest number of points) computes the eigenvalue to the desired accuracy. It is probable that if something other than the eigenvalue, e.g. the eigenfunction only or the expection of operators, were the primary objective, another strategy may be preferred.

5.3.2.1 Equidistribution Strategy

Pereyra and Sewell[33] suggest a strategy in which $\tau(x_i)$, i=1,2,...,N, is approximately constant. This is the strategy used by Lentini and Pereyra to solve their first order systems of boundary value problems. Implementation of such a strategy is quite easy. We simply split any point x_i at which $|T(x_i)|$ is greater than some tolerance, EPS. We then resolve the matrix problem on the new mesh and repeat. We know from Theorem 5.5 that eventually all $|\tau(x_i)|$ will be less than EPS. We have already argued that this process will also decrease $|\Delta\lambda|$. We show some numerical results using this strategy after all strategies have been considered.

5.3.2.2 Solution-Weighted Strategy

Expression (4.3.21) gave us an error estimate for $\Delta\lambda$. We can also get a less accurate estimate by the following reasoning.

We assume now that an accurate eigenvalue is our objective and that the error in the eigenfunction is unimportant compared to the error in the eigenvalue. Recalling now that

$$A\bar{y} - \lambda B\bar{y} = \bar{\tau}(\bar{y}) \qquad (5.3.4)$$

we let $\lambda = \Lambda + \Delta\lambda$ and, by our assumption, we let $\bar{y} = Y$. Substituting into (5.3.4) and multiplying by Y^t we get

$$Y^t(AY - \Lambda BY) - \Delta\lambda Y^t BY \approx Y^t\,\bar{\tau}(\bar{y}).$$

The first term is zero by expression (4.3.2) and so we see that

$$\Delta\lambda \approx -\,Y^t\,\bar{\tau}(\bar{y}) \;/\; Y^t BY.$$

The numerator can be written as

$$-\,Y^t\,\bar{\tau}(\bar{y}) = -\sum_{i=0}^{N+1} Y_i\,\tau(x_i),$$

and so from this estimate of $\Delta\lambda$ it is obvious that, while making $|\tau(x_i)|$ small will decrease $|\Delta\lambda|$, a better strategy should be to make $|Y_i \tau(x_i)|$ small.

So our second strategy is to split any point for which

$$|Y_i \; T(x_i)| > \text{EPS}$$

thus leading to final mesh on which

$$|Y_i \tau(x_i)| \approx |Y_j \tau(x_j)| \text{ for all } i,j.$$

5.3.2.3 <u>D^{-1} Solution-Weighted Strategy</u>

Again considering the error estimate (4.3.22)

$$\Delta\lambda \approx -Y^t D^{-1} D^{-1} \bar{\tau}(\bar{y}) \,/\, Y^t D^{-1} D^{-1} BY \qquad (5.3.5)$$

we introduce our third strategy. The numerator of (5.3.5) is

$$-\sum_{i=0}^{N+1} d_{ii}^{-2} \, Y_i \; \tau(x_i) \qquad (5.3.6)$$

where the $\{d_{ii}\}$ are elements of D. If this expression can be made small then $|\Delta\lambda|$ will also be small.

The first two strategies do in fact make (5.3.5) smaller. However, expression (5.3.6) implies that what should actually be equidistributed is $d_{ii}^{-2} Y_i \tau(x_i)$. Thus our third strategy is to split any point x_i for which $|d_{ii}^{-2} Y_i \; T(x_i)| > \text{EPS}$. This will lead to a mesh on which

$$|d_{ii}^{-2} Y_i \; \tau(x_i)| \approx |d_{jj}^{-2} Y_j \; \tau(x_j)| \text{ for all } i,j.$$

Implementation of this strategy is slightly more difficult because the matrix D^{-1} would not normally be computed. We must recall from Section 4.3 that the coefficients α_{0i}, β_{0i}, α_{2i}, and β_{2i}, that make up d_{ii}, must be recomputed for each new mesh.

In Table 5.5 we compare the 3 strategies by using each to solve several problems. The value of EPS is held constant. We show the number of points (N) required and the error in the eigenvalue ($|e|$) resulting in each case.

The results in Table 5.5 are but a few of those we computed but they are quite typical of the observed phenomena. The equidistribution strategy uses the most points in almost every case, however, the errors are quite similar. The other 2 strategies tend to be indistinguishable when solving the regular problems of this chapter.

This can best be explained by considering an example. We look at the problem

$$y'' = -\lambda y, \quad y(0) = y(1) = 0$$

which generated the first row of Table 5.5. The final mesh of 78 intervals consists of only 2 different sizes of h. Either $h = 1/64$ or $h = 1/128$. This relatively smooth mesh produces a matrix $(D^{-1})^2$ which has a 1 at 37 of its 79 diagonal elements. The other elements are either 1.5 or 2. Thus the inclusion of the terms d_{ii}^{-2} into the splitting strategy has very little effect on the final mesh when the eigenfunctions are well-behaved like those of this chapter.

Table 5.5

Comparison of Refinement Strategies

	Equidist		SW		D^{-1} - SW	
q(x)	N	$\|e\|$	N	$\|e\|$	N	$\|e\|$
0	158	1.56D-7	78	6.43D-8	78	6.43D-8
6cos(2x)	126	9.23D-8	102	1.20D-6	98	1.98D-7
16cos(2x)	66	6.82D-6	54	2.84D-5	58	5.06D-6
x|x|	38	5.86D-7	28	6.91D-6	34	1.19D-6

To make a decision on which strategy to use we look at the error estimates which motivated the 2 strategies. Using the same problem as an example, we find that for both strategies $|e| \approx 6.430 \times 10^{-8}$, but $|e_{est}|$ for the SW strategy is about 1.163×10^{-7} while for the D^{-1} - SW, $|e_{est}| \approx 6.428 \times 10^{-8}$. Similar differences are observed for all the problems looked at. Thus we conclude that when accurate computation of the eigenvalue is our primary objective, the D^{-1} solution-weighted strategy should be used. This is the strategy used to obtain all further results of this chapter and of Chapter 7.

5.3.3 The Adaptive, Iterative Algorithm

We now present the adaptive, iterative algorithm.

ALGORITHM 5.6:

1. Set up a basic uniform mesh, π_0, on [a,b].
2. Solve the matrix problem (4.3.2) using Algorithm 4.5.

3. Compute error estimate (4.3.22) and make one correction using Algortihm 4.21.
4. If $|\overline{\Delta\lambda}| < \text{TOL}$, STOP.
5. Refine mesh:
   i) Split all points x_i for which

   $$|d_{ii}^{-2}\, Y_i\, T(x_i)| > \text{EPS}.$$

   (See below for a discussion of how to compute EPS from a given TOL.)

   ii) Interpolate values of Y_i at new mesh points to get starting right-hand side for next iteration.
6. GO TO 2.

In Step 5 we mention a relationship between TOL (the accuracy desired in λ) and EPS (the equidistribution level at each point of the mesh). It is unreasonable to expect the user of this algorithm to specify EPS, though he must specify TOL. The EPS required by the splitting strategy can be computed for each iteration.

We want

$$|\Delta\lambda| \approx \frac{|Y^t D^{-1} D^{-1} T(Y)|}{|Y^t D^{-1} D^{-1} BY|} < \text{TOL}.$$

This will be accomplished if

$$|Y^t D^{-1} D^{-1} T(Y)| < \text{TOL} \cdot |Y^t D^{-1} D^{-1} BY| \qquad (5.3.7)$$

but $$|Y^t D^{-1} D^{-1} T(Y)| \leq \sum_{i=0}^{N+1} |d_{ii}^{-2}\, Y_i\, T(x_i)|.$$

Since each $|d_{ii}^{-2} Y_i T(x_i)|$ is weighted equally, (5.3.7) will be satisfied if

$$|d_{ii}^{-2} Y_i T(x_i)| < \frac{1}{(N+2)} \cdot \text{TOL} \cdot |Y^t D^{-1} D^{-1} BY| .$$

Thus if we let

$$\text{EPS} = \frac{1}{(N+2)} \cdot \text{TOL} \cdot |Y^t D^{-1} D^{-1} BY| \qquad (5.3.8)$$

the tolerance will be met. In practice, however, we found that this choice of EPS is too small. Using it the actual $|\Delta\lambda|$ turns out, in many cases, to be much less than TOL thus causing extra work. We attribute this to at least the following phenomena.

1. The terms of $Y^t D^{-1} D^{-1} \bar{\tau}(\bar{y})$ are both positive and negative. Thus there is a considerable amount of cancellation occurring in the actual $\Delta\lambda$.

We have found that the following heuristic gives better results.

a. Count the number of times $\text{POS} = Y_i T(x_i) > 0$ and $\text{NEG} = Y_i T(x_i) < 0$.

b. Instead of N+2 in (5.3.8) use $|\text{POS} - \text{NEG}|$. If $|\text{POS} - \text{NEG}| = 0$ we use 1.

Our reasoning is that if all $|d_{ii}^{-2} Y_i \tau(x_i)| = \text{EPS}$ then POS and NEG would cancel out in $Y^t D^{-1} D^{-1} \bar{\tau}(\bar{y})$ and only the excess one way or the other would affect $\Delta\lambda$.

2. If, on iteration j, $|d_{ii}^{-2} Y_i T(x_i)|$ is only slightly larger than EPS, we find that splitting nodes makes it much less on iteration j+1. This tends to make

the final $\Delta\lambda$ too small.

This is a difficult occurrence to prevent because there is no way to foresee the exact effect splitting will have since it depends on all points of the mesh and their interaction.

One possible solution would be to let

$$\text{EPS} = C \cdot \frac{1}{|\text{POS} - \text{NEG}|} \cdot \text{TOL} \cdot |Y^t D^{-1} D^{-1} BY|$$

where C is some constant > 1. Then if we reach a point in the iteration where $|\Delta\lambda| >$ TOL but all $|d_{ii}^{-2}\, Y_i\, T(x_i)| <$ EPS, we decrease C. This process could be quite inefficient if C were not decreased by the proper amount.

We do not use such a "sliding" EPS in our test programs. Rather we are satisfied with an actual $|\Delta\lambda|$ smaller than the requested TOL.

5.3.4 Numerical Results

We wish to compare the performance of Algorithm 5.6 with that of the uniform mesh method of Section 4.2. We tested our algorithm on each of the following problems with the results shown below. After the algorithm found its optimal N we ran the Numerov method using the same N and the 2 closest powers of 2. This was done because the non-adaptive method of Section 5.2 is restricted to N being a power of 2.

Problems I and II are described in Section 5.2.4.

<u>Problem III</u>: (Weber's Equation)

$y'' (\lambda - x^2)y = 0$

$y(0) = y(1) = 0$

$\lambda_0 \approx 10.1511640305$

$\lambda_4 \approx 247.0715002280$

<u>Problem IV</u>: (Mathieu's Equation)

$y'' + (\lambda - 2q\cos 2x)y = 0$

$y(0) = y(\pi) = 0$

<u>$q = 1$</u>

$\lambda_0 \approx -.1102488$

$\lambda_4 \approx 25.0208408$

<u>$q = 3$</u>

$\lambda_1 \approx 3.2769220$

$\lambda_3 \approx 16.2727010$

<u>$q = 8$</u>

$\lambda_0 \approx -10.6053681$

$\lambda_4 \approx 26.2209995$

While the results shown in Table 5.6 do display some advantages of the adaptive method, we find it necessary to point out that the method does not perform as well on regular problems as on the singular problems of Chapter 7.

An adaptive strategy seems much better suited to problems in which the solution y(x) has singularities or long, exponentially decaying "tails" rather than ones in which the solution is more well-behaved.

Table 5.6

Results of Algorithm 5.6

Problem	Method	N	$\|e\|$	$\|e_{est}\|$	$\|e_c\|$
I, λ_0	adaptive	78	6.430D-8	6.426D-8	3.706D-11
	Numerov	64	2.391D-7	2.387D-7	7.006D-11
		78	1.082D-7	1.082D-7	2.146D-11
		128	1.492D-8	1.492D-8	1.258D-12
I, λ_3	adaptive	136	4.404D-5	4.397D-5	6.615D-8
	Numerov	128	6.115D-5	6.108D-5	7.109D-8
		136	4.798D-5	4.793D-5	4.957D-8
		256	3.821D-6	3.819D-6	1.130D-9
II, λ_0	adaptive	74	*6.5 D-8	2.225D-8	*4.3 D-8
	Numerov	64	*6.7 D-8	5.388D-8	*1.4 D-8
		74	*4.3 D-8	3.015D-8	*1.3 D-8
		128	*1.5 D-8	3.367D-9	*1.2 D-8
II, λ_3	adaptive	140	9.368D-6	9.306D-6	*6.13 D-8
	Numerov	128	1.533D-5	1.527D-5	*6.18 D-8
		140	1.073D-5	1.067D-5	*5.45 D-8
		256	9.996D-7	9.551D-7	*4.45 D-8
III, λ_0	adaptive	37	1.719D-6	1.714D-6	*5.3 D-9
	Numerov	32	4.137D-6	4.131D-6	*6.3 D-9
		37	2.314D-6	2.312D-6	*2.7 D-9
		64	2.585D-7	2.584D-7	*1. D-10
III, λ_4	adaptive	147	4.007D-4	4.007D-4	1.18 D-8
	Numerov	128	2.333D-4	2.329D-4	4.197D-7
		147	1.341D-4	1.339D-4	1.848D-7
		256	1.458D-5	1.457D-5	*6.7 D-9

Table 5.6 (continued)

Problem	Method	N	$\lvert e \rvert$	$\lvert e_{est} \rvert$	$\lvert e_c \rvert$
IV, λ_0	adaptive	70	3.719D-7	3.546D-7	*1.7 D-8
q = 1	Numerov	64	2.601D-7	2.426D-7	*1.7 D-8
		70	1.868D-7	1.696D-7	*1.7 D-8
		128	*3.2 D-8	1.518D-8	*1.7 D-8
IV, λ_4	adaptive	152	1.03 D-5	1.029D-5	*1. D-8
q = 1	Numerov	128	2.38 D-5	2.378D-5	*2. D-8
		152	1.20 D-5	1.196D-5	*1. D-8
		256	1.47 D-6	1.487D-6	*1. D-8
IV, λ_1	adaptive	112	*3.2 D-8	6.888D-8	*3. D-8
q = 3	Numerov	64	7.47 D-6	7.414D-6	*6.1 D-8
		112	8.23 D-7	7.918D-7	*3.3 D-8
		128	4.95 D-7	4.642D-7	*3.1 D-8
IV, λ_3	adaptive	128	8.640D-6	8.816D-6	1.8 D-7
q = 3	Numerov	128	7.100D-6	7.283D-6	1.8 D-7
IV, λ_0	adaptive	114	5.5 D-7	2.511D-8	*5. D-8
q = 8	Numerov	64	6.9 D-6	6.827D-6	*8. D-8
		114	7.2 D-7	6.800D-7	*4. D-8
		128	4.6 D-7	4.280D-7	*4. D-8
IV, λ_4	adaptive	164	2.057D-5	2.050D-5	*7. D-8
q = 8	Numerov	128	3.560D-5	3.547D-5	1.4 D-7
		164	1.322D-5	1.317D-5	*5. D-8
		256	2.25 D-6	2.220D-6	*3. D-8

* Correct to all available digits

Even with these concessions we find that in half of the data sets, $|e|$ is smaller with the adaptive method than with the Numerov method using the same N. More significantly we find that if a non-adaptive method like that of Section 5.2 were used, we would need to go to the next highest power of 2 in almost every case to get equivalent or better accuracy.

Another result of Table 5.6 is the confirmation of our analysis concerning the asymptotic order of convergence. Theorem 4.19 assures us that $\Lambda - \lambda = O(h^4)$ and the results of Section 4.3.7 show that $\Lambda + \overline{\Delta\lambda} - \lambda = O(h^6)$.

During the running of Algorithm 5.6, N proceeds from a value of 8, through several intermediate values, to its final value as shown in Table 5.6.

If $|e|$ and $|e_c|$ are computed when $N = 8$ (call them $|e|_8$ and $|e_c|_8$) then the orders of convergence can be computed by solving the following equations:

$$\frac{|e|_8}{(I/8)^{\alpha}} = \frac{|e|}{(I/N)^{\alpha}}$$

and
$$\frac{|e_c|_8}{(I/8)^{\beta}} = \frac{|e_c|}{(I/N)^{\beta}} \quad .$$

If the theoretical results mentioned above are correct then α should be 4 and β should be 6. Using Problem I as our example, we find that for λ_0 we must solve

$$\frac{9.839\text{x}10^{-4}}{(\pi/8)^{\alpha}} = \frac{6.430\text{x}10^{-8}}{(\pi/78)^{\alpha}}$$

and
$$\frac{5.125\text{x}10^{-5}}{(\pi/8)^{\beta}} = \frac{3.706\text{x}10^{-11}}{(\pi/78)^{\beta}}$$

to get $\alpha = 4.23$ and $\beta = 6.21$. For λ_3, $\alpha = 4.05$ and $\beta = 5.89$. Similar values are obtained from the other problems when enough decimal places are available in the exact λ.

Figures 5.5 and 5.6 show the distribution of the selected points for λ_0 and λ_4 of Problem III. We see from these graphs that only a few different sizes of h are used to solve a regular problem. As we mentioned above, however, it is the placement of the points which makes the adaptive method more advantageous. More dramatic step size changes are evident in the singular problems of Chapter 7.

In the figures, the points along the x-axis are the $\{x_i\}$ placed by the algorithm. The graph itself shows the eigenfunction computed by the method. In Figure 5.6 only half of the eigenfunction is shown. The function is, however, symmetric about x = .5.

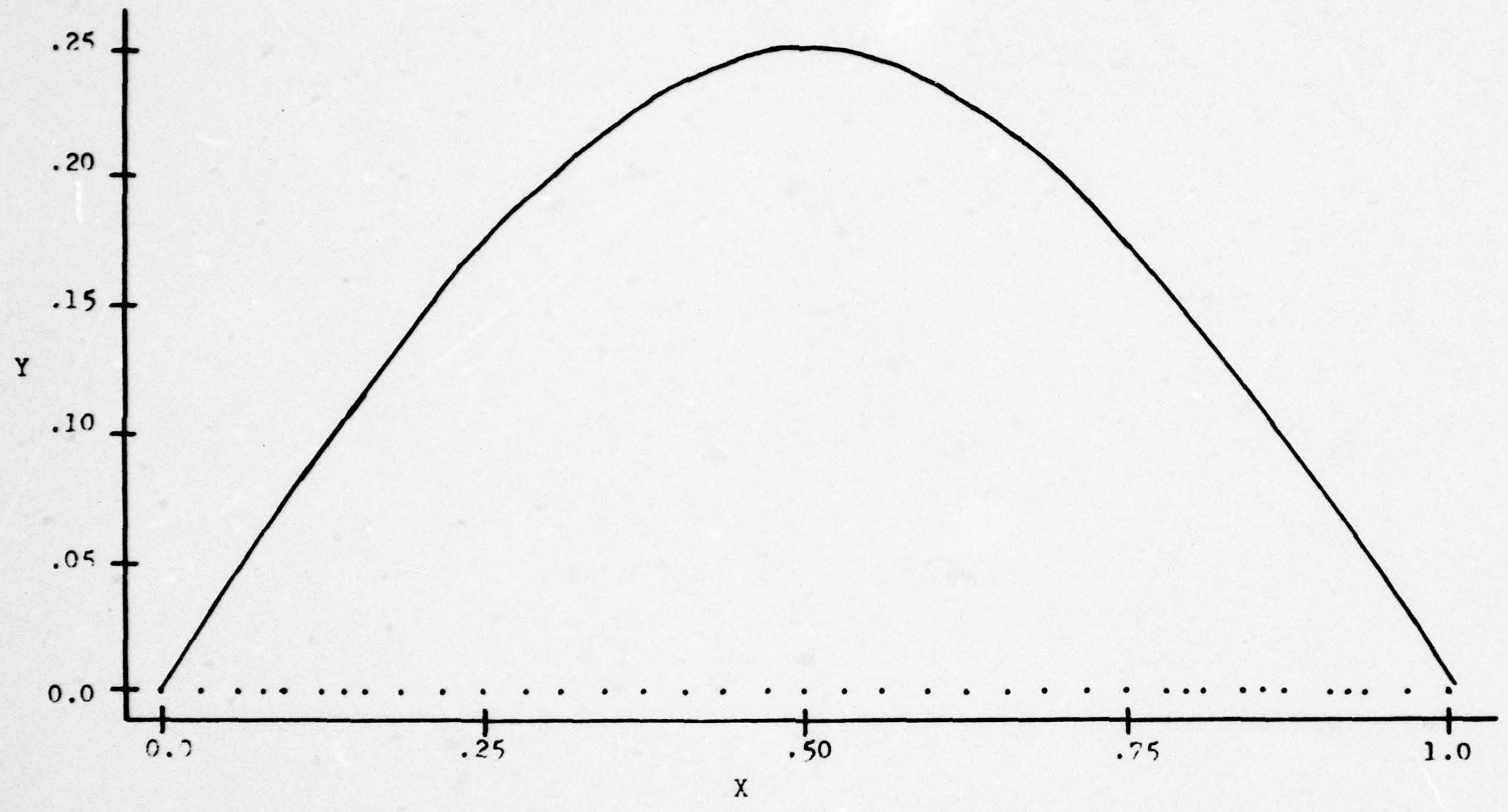


Figure 5.5

Mesh Distribution for Problem III, λ_0

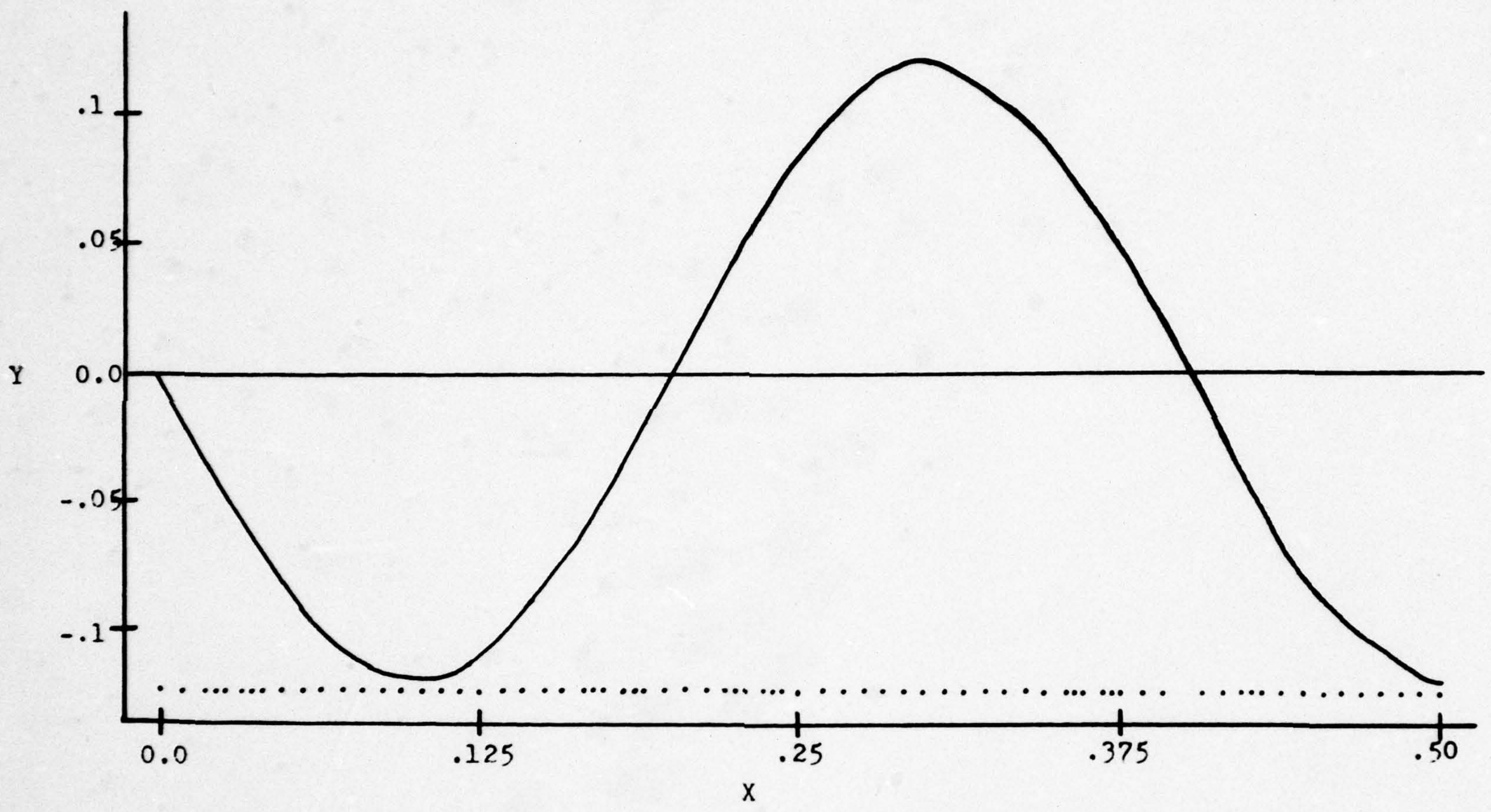


Figure 5.6

Mesh Distribution for Problem III, λ_4

Chapter 6

STARTING VALUE AND RECYCLING PROBLEMS

6.1 Starting Value Problem

In the algorithms of Chapters 4 and 5 it was assumed that a good estimate, $\Lambda_{i,0}$, to the desired eigenvalue, λ_i (or Λ_i), was always available.

In some applications it is possible to get such an estimate. It may be obtained through theoretical means by comparing the given problem with a known problem, or when solving a specific problem which represents a physical process, an estimate may be obtained through experimental results. When a good approximation is supplied it is evident that convergence of the adaptive algorithm will occur quite quickly and the computer program will be highly efficient.

In many other applications the user cannot supply an estimate to the eigenvalue. Instead he knows only that he wants the kth smallest eigenvalue, λ_k, or equivalently the eigenvalue corresponding to $y_k(x)$, the eigenfunction with exactly k zeros in (a,b). Our objective then in this section is to develop an algorithm for computing $\Lambda_{i,0}$.

6.1.1 Objectives of the Algorithm

We believe the following objectives to be the most important in developing a starting value algorithm. Thus we consider each in our discussion.

1. The algorithm must be efficient. The entire advantage of our adaptive algorithm can be lost if there is no efficient method for finding a starting value.

We argue in Chapter 4 that our non-uniform finite-difference method gives cubic convergence, while the shooting method algorithms like the one by Bailey, Gordon and Shampine (see Chapter 3 for a discussion) give only quadratic convergence. However, Bailey, et al., claim that their algorithm will converge to the proper eigenvalue regardless of the starting value. This, of course, is because the number k (defined above) is a parameter in the boundary conditions. Since our method uses inverse iteration it will converge to the eigenvalue closest to the initial value. We must therefore use a procedure which computes a starting value without substantially reducing the convergence rate.

2. The algorithm should be no more accurate than necessary. This idea is discussed in greater detail in Section 6.1.3. We show there that the starting value algorithm converges much more slowly than the adaptive algorithm and thus we do not want it to operate longer than necessary.

3. The algorithm must use existing techniques. By this we mean that the algorithm should be designed to use as many of the same procedures as the main adaptive algorithm as possible.

While it might be possible to use a shooting method, for example, to get a starting value, it would mean introducing a whole new set of procedures. We instead find a starting value directly from the matrix problem (4.3.2).

6.1.2 Sturm Sequences

Let us assume that the user of our adaptive S-L algorithm supplies us with the index of the eigenvalue desired, i.e. k is the input when λ_k is being requested. We wish to show that an appropriate choice for an approximation to λ_k is an estimate, $\Lambda_{k,0}$, of Λ_k, the kth matrix eigenvalue.

We recall from Chapter 4 that the eigenvalues of the S-L problem are approximated by the eigenvalues of the matrix problem

$$(A - \Lambda B)Y = 0. \qquad (6.1.1)$$

We know from Theorem 4.19 that for every eigenvalue, λ_k, of the differential equation there exists an eigenvalue, Λ, of the matrix problem (6.1.1) such that

$$(\Lambda - \lambda_k) = O(h^4) \quad \text{as } h \to 0.$$

To get convergence in the other direction we use Theorem 3.2 and Lemma 4.1 of [29]. Because our discretization operator, (4.3.1), call it L_h, is such that

$$\lim_{h \to 0} L_h u = L u \qquad (6.1.2)$$

where L is the normal form of the regular S-L operator, we can say that spectrum(L_h) $\to$ spectrum(L) as $h \to 0$.

To solve singular S-L problems we truncate the interval (a,b) to a finite interval in which q is continuous. In order for the limit (6.1.2) to hold as $h \to 0$ and as the truncated interval $\to$ (a,b) we must require that $q \in C(a,b)$.

We recall from Chapter 2 that singular problems can have certain regions of continuous spectrum. We must further restrict our class of problems to those in which any continuous spectrum is greater than the point spectrum.

Using these facts we can now prove the following.

<u>THEOREM 6.1</u>: Let k be fixed and less than or equal to the number of eigenvalues in the point spectrum of (4.1.1) and let any continuous spectrum of L be greater than the point spectrum. Then as $h \to 0$, Λ_j of the system (6.1.1) converges to λ_j of (4.1.1) for all $j = 0,1,\ldots,k$.

<u>Proof</u>: We require k to be fixed because we do not want it to go to ∞ as $h \to 0$. The theorem holds, however, for any choice of k in the proper range.

First we recall from Chapter 2 that the λ_i of the point spectrum can be ordered as follows:

$$\lambda_0 < \lambda_1 < \lambda_2 < \ldots$$

We assume that the Λ_i are also ordered as follows:

$$\Lambda_0 < \Lambda_1 < \ldots < \Lambda_{N-1}.$$

Theorem 4.16 assures us that for h small enough the Λ_i are distinct. In addition there are only a finite number of Λ_i and thus the above ordering is possible.

We now prove our theorem by induction. We start by showing that as $h \to 0$, $\Lambda_0 \to \lambda_0$.

We know that there exists some j for which $\Lambda_j \to \lambda_0$. Let us assume that Λ_j is not the smallest matrix eigenvalue, i.e. $j \neq 0$. Then $\Lambda_0 \to$ some $\lambda_m > \lambda_0$.

Define $\lambda_m - \lambda_0 = \delta > 0$. We let h be small enough so that

$$|\Lambda_j - \lambda_0| < \delta/2 \quad \text{and} \quad |\Lambda_0 - \lambda_m| < \delta/2.$$

This implies that

$$\lambda_0 - \delta/2 < \Lambda_j < \lambda_0 + \delta/2 = \frac{\lambda_0 + \lambda_m}{2}$$

and
$$\frac{\lambda_0 + \lambda_m}{2} = \lambda_m - \delta/2 < \Lambda_0 < \lambda_m + \delta/2$$

which in turn imply that $\Lambda_j < \Lambda_0$. This is a contradiction to the ordering of the Λ_j and thus $\Lambda_0 \to \lambda_0$.

Let us now assume that $\Lambda_j \to \lambda_j$, $j=0,1,\ldots,k-1$, and show that $\Lambda_k \to \lambda_k$. We know that there exists a p such that $\Lambda_p \to \lambda_k$. From our induction hypothesis we know that $p \neq 0,1,\ldots,k-1$. We assume that $p \neq k$, i.e. $p > k$. It is true, however, that $\Lambda_k \to$ some $\lambda_m < \lambda_k$. Thus we have

$$\Lambda_p \to \lambda_k$$
$$\Lambda_k \to \lambda_m < \lambda_k$$
$$\Lambda_k < \Lambda_p.$$

Letting $\lambda_m - \lambda_k = \delta > 0$, we can choose h small enough such that

$$|\Lambda_p - \lambda_k| < \delta/2$$

and $$|\Lambda_k - \lambda_m| < \delta/2.$$

Together these imply that $\Lambda_p < \Lambda_k$ which is a contradiction to the ordering of the Λ_j. Thus the theorem is proved.

This theorem tells us that as h becomes small, λ_k will be approximated by Λ_k, the k<u>th</u> smallest matrix eigenvalue. Thus as a starting value, $\Lambda_{k,0}$, for Algorithm 5.6 we approximate the k<u>th</u> eigenvalue of the original matrix problem.

The procedure we use to get this approximation is based on the Sturm sequence property of the matrix problem (6.1.1). We use a bisection method described by Wilkinson [42] to locate and approximate the k<u>th</u> eigenvalue.

Define the following sequence of functions derived from the matrix problem (6.1.1).

$$p_0(\mu) = 1$$

$$p_1(\mu) = \det\,(A - \mu B)_1$$

$$= \alpha_{11} - \beta_{11}(\mu - q_1)$$

$$\vdots$$

$$p_i(\mu) = \det\,(A - \mu B)_i$$

$$= \{\alpha_{1i} - \beta_{1i}(\mu - q_i)\}\, p_{i-1}(\mu)$$
$$- \{\alpha_{2,i-1} - \beta_{2,i-1}(\mu - q_i)\} \cdot$$
$$\{\alpha_{0i} - \beta_{0i}(\mu - q_{i-1})\}\, p_{i-2}(\mu)$$
$$i=2,3,\ldots,N.$$

If we let the sequence be evaluated for some μ_0 and let $s(\mu_0)$ denote the number of sign agreements of consecutive members of the sequence, we can use the following result proved by Wilkinson[42, p. 300]:

The value $s(\mu_0)$ is the number of eigenvalues of $(A - \mu B)Y = 0$ which are strictly greater than μ_0.

This result is, of course, only valid if the sequence $p_0, p_1, \ldots, p_N$ is a Sturm sequence. However, we know from the proof of Theorem 4.16 that there does exist an h such that for $\max_i h_i = h$, the sequence is a Sturm sequence. We have found, however, in all our test problems that even the coarsest mesh, $n = 8$, yields real eigenvalues which are reasonable estimates to λ_k. Therefore in most cases there does not appear to be a need to decrease h to ensure that we have a Sturm sequence.

One check must be made, however, which may lead to increasing N immediately. If the user asks for λ_k where $k > N$, then the matrix problem cannot supply us with the proper eigenvalue. Thus N must be increased even before the adaptive algorithm runs. This check, therefore, is included as part of Algorithm 6.4.

6.1.3 Bisection Algorithm

We use the following algorithm to locate the kth eigenvalue of (6.1.1).

ALGORITHM 6.2:

1. Let $b_0 > a_0$ be such that $s(a_0) \geq N-k$ and $s(b_0) < N-k$. (We then know that Λ_k lies in the interval (a_0, b_0).)
2. Let $i = 1$.
3. Let $c_i = (a_{i-1} + b_{i-1})/2$.
4. Compute $p_0(c_i), p_1(c_i), \ldots, p_N(c_i)$.
5. Compute $s(c_i)$.
6. If $s(c_i) \geq N-k$ take $a_i = c_i$ and $b_i = b_{i-1}$. If $s(c_i) < N-k$ take $a_i = a_{i-1}$ and $b_i = c_i$.
7. If $(b_i - a_i) < \epsilon$ then STOP with $\Lambda_{k,0} = c_i$ else increment i by 1 and GO TO 3.

This process can continue until $\Lambda_{k,0}$ is as accurate as desired. We see, however, that the algorithm only yields linear convergence to Λ_k because $(b_i - a_i) = (b_{i-1} - a_{i-1})/2$. Therefore we must only continue long enough to ensure convergence to the correct eigenvalue.

By the nature of inverse iteration, this convergence is only made certain if our estimate, $\Lambda_{k,0}$, is closer to Λ_k than to any other eigenvalue. If this were not the case we would get convergence to the closest, and thus the wrong, eigenvalue. The size of the interval (a_i, b_i) is thus dependent on how close the eigenvalues are.

We estimate Λ_k by $c_{i+1} = (a_i + b_i)/2$. We know that Λ_k lies in the interval (a_i, b_i) and so

$$|c_{i+1} - \Lambda_k| < (b_i - a_i)/2.$$

If we require also that $b_i \le \Lambda_{k+1}$ and $a_i \ge \Lambda_{k-1}$ or equivalently that $s(b_i) = N-k-1$ and $s(a_i) = N-k$ then we see that

$$|c_{i+1} - \Lambda_j| \ge (b_i - a_i)/2, \quad j \ne k$$

and thus c_{i+1} is closest to Λ_k. This check is quite easy to implement because $s(a_i)$ and $s(b_i)$ have already been computed when $s(c_{i+1})$ is being evaluated.

One final point to be considered before presenting our algorithm is the problem of finding appropriate a_0 and b_0. Wilkinson[42] notes that suitable a_0 and b_0 when $B = I$ are $\pm \|A\|_\infty$. This, of course, is a consequence of the well known result

$$\rho(A) \le \max_{1 \le i \le N} \sum_{j=1}^{N} |a_{ij}|$$

(Varga[41], p. 17) where $\rho(A)$ is the spectral radius of A.

For our problem a_0 and b_0 can be computed by noting the following result.

<u>THEOREM 6.3</u>: Let the generalized eigenvalue problem, $(A - \Lambda B)Y = 0$, be such that B is strictly diagonally dominant. Then

$$\max_k |\lambda_k| \le \{ \sum_{j=1}^{N} |a_{ij}| / (|b_{ii}| - \sum_{\substack{j=1 \\ j \ne i}}^{N} |b_{ij}|) \}$$

where i is such that $|Y_i| \geq |Y_j|$, $j \neq i$.

<u>Proof</u>: Let λ be any eigenvalue of the generalized problem and let Y be an eigenvector corresponding to λ. Let Y_i be the largest component, in absolute value, of Y, i.e. $|Y_i| \geq |Y_j|$, $j \neq i$.

Writing the ith equation we have

$$\sum_{j=1}^{N} (a_{ij} - \lambda b_{ij}) Y_j = 0$$

or

$$(b_{ii}\lambda - a_{ii}) Y_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} (a_{ij} - \lambda b_{ij}) Y_j .$$

So by the triangle inequality we find that

$$|b_{ii}\lambda - a_{ii}|\,|Y_i| \leq \sum_{j \neq i} |a_{ij} - \lambda b_{ij}|\,|Y_j| .$$

Dividing by $|Y_i|$ yields

$$|b_{ii}\lambda - a_{ii}| \leq \sum_{j \neq i} |a_{ij} - \lambda b_{ij}|\,|Y_j| / |Y_i|$$

$$\leq \sum_{j \neq i} |a_{ij} - \lambda b_{ij}| .$$

Using the triangle equality on both sides we get

$$|b_{ii}|\,|\lambda| - |a_{ii}| \leq \sum_{j \neq i} |a_{ij}| + |\lambda| \sum_{j \neq i} |b_{ij}|$$

or

$$|\lambda| \{ |b_{ii}| - \sum_{j \neq i} |b_{ij}| \} \leq \sum_{j=1}^{N} |a_{ij}| .$$

Since $b_i = |b_{ii}| - \sum_{j \neq i} |b_{ij}| > 0$

we have finally

$$|\lambda| \leq \sum_{j=1}^{N} |a_{ij}| / b_i$$

and the theorem is proved.

Since we do not know which component of Y is largest before Y is found we use

$$\pm \max_i \{ \sum_{j=1}^{N} |a_{ij}| / b_i \} \tag{6.1.3}$$

for a_0 and b_0.

6.1.4 Starting Value Algorithm

We now present the starting value algorithm.

ALGORITHM 6.4: Let N = 8. (arbitrary)

```
1. If $k > N$ then $N = 2N$ and repeat 1.
2. Set up matrices A and B.
3. Compute $a_0$ and $b_0$ from (6.1.3).
4. Compute $\Lambda_{k,0}$ by Algorithm 6.1 with the following exception:
      At Step 7 we have
         7. If $s(a_i) = N-k$ and $s(b_i) = N-k-1$
            then $\delta = (b_i - a_i)/2$ and STOP with
            $\Lambda_{k,0} = c_{i+1}$ else $i=i+1$ and GO TO 3.
```

It should be noted that when Algorithm 6.4 is completed the matrix $A - \Lambda_{k,0}B$ is exactly the one we require to execute Algorithm 5.6. Thus all our objectives of Section 6.1.1 have been satisfied.

6.1.5 Numerical Results

We present here 2 examples of the starting values found by Algorithm 6.3. The problems chosen are Problems I and III of Chapter 5. These problems were selected because many exact eigenvalues are known and our algorithm can be well tested.

In Tables 6.1 and 6.2 we show several eigenvalues, λ_i, followed by the estimate, EV, from Algorithm 6.4. The final column shows the number of bisections needed to get the value.

In each case, the estimated value is closer to the ith eigenvalue than any other. We also note that as i gets larger, the number of iterations decreases. This is, of course, due to the fact that in these problems the eigenvalues are more greatly separated as i gets larger and thus not as many bisections are necessary to ensure convergence to the proper eigenvalue.

Table 6.1

Results of Algorithm 6.3 on Problem I

i	λ_i	EV	# bisect
0	9.870	6.0	10
1	39.478	42.0	10
2	88.826	84.0	9
3	157.914	168.0	8
4	246.740	264.0	8
5	355.306	360.0	8
10	1194.22	1200.0	7
19	3947.84	3942.0	6

Table 6.2

Results of Algorithm 6.3 on Problem III

i	λ_i	EV	# bisect
0	10.151	6.001	10
1	39.799	42.006	10
2	89.154	84.012	9
3	158.244	168.024	8
4	247.072	264.038	8
5	355.638	360.051	8
10	1194.555	1200.172	7
19	3948.175	3744.535	6

6.2 Recycling Problem

We now consider a problem closely related to the starting value problem of Section 6.1. We call it the <u>Recycling Problem</u> for reasons which will be obvious.

Recalling Algorithm 5.6, the adaptive, iterative algorithm, we note that after each iteration of the algorithm we must set up a new matrix in order to begin the next iteration. This requires a value of Λ (call it $\Lambda_{k,i}$ for the <u>ith</u> iteration of Λ_k) to set up the matrix $A - \Lambda_{k,i}B$.

A first inclination is to use the value of Λ found after iteration i-1 was completed. We see from the following numerical example, however, that this can lead to incorrect answers.

The problem we solve is a singular S-L problem:

$$y'' = \left(-\frac{1}{x} - \lambda\right)y$$

$$y(0) = y(\infty) = 0$$

having eigenvalues $\lambda_n = -1/(4n^2)$, $n=1,2,\ldots$ Such a problem is much more likely to converge to a wrong eigenvalue than is a regular problem.

We attempt to compute $\lambda_2 = -.0625$ using the technique described above. Table 6.3 shows the values of and $\bar{\lambda}$ as N increases. Recall that $\bar{\lambda}$ is the corrected eigenvalue. Algorithm 6.4 was used to compute a starting value of $\Lambda = -.03721$. Convergence to the wrong eigenvalue has occurred.

Table 6.3

Recycling of Λ_i

N	Λ_i	$\bar{\lambda}$
starting value	-.03721	
8	-.03890	-.03934
18	-.04101	-.04474
25	-.05910	-.25541
30	-.14547	-.22907
42	-.23886	-.29142
51	-.24418	-.24553
55	-.24699	-.24704

During early iterations of Algorithm 5.6, when the mesh is coarse, the kth matrix eigenvalue may be quite different from the exact λ_k. It may even be closer to some other λ_j. This is seen in the numerical example of Table 6.3. This phenomenon is especially prevalent when higher eigenvalues are desired.

A second possible solution is to run the starting value algorithm before each iteration of Algorithm 5.6. This assumes no knowledge of the eigenvalue. In later iterations we can certainly expect that the recycled value will be quite close to the desired eigenvalue. Thus this second strategy would tend to be quite inneficient.

What we need then is a procedure to check $\Lambda_{k,i-1}$ for validity, i.e. whether it falls within $(\Lambda_{k-1}, \Lambda_{k+1})$ and whether it is closest to Λ_k. If these conditions are not met, the procedure must produce a value for which they are.

We claim that the following algorithm will fulfill these requirements. In Steps 4.1 and 5.1 the number δ is used. The reader is referred back to Algorithm 6.4 to find that $\delta = (b_i - a_i)/2$, i.e. half the length of the interval found to contain $\Lambda_{k,0}$. The algorithm attempts to show that the interval $(\Lambda_{k,i-1} - \delta, \Lambda_{k,i-1} + \delta)$ lies within $(\Lambda_{k-1}, \Lambda_{k+1})$. If so then $\Lambda_{k,i-1}$ can serve as the starting value for iteration i. If this cannot be shown then a new value must be found.

<u>ALGORITHM 6.5</u>: Let $\Lambda_{k,i-1}$ be the matrix eigenvalue computed by iteration i-1 of Algorithm 5.6.

```
1. Set up the matrix $A - \Lambda_{k,i-1}B$ for iteration i.
2. Compute $s_k = s(\Lambda_{k,i-1})$.
3. If:
      $s_k = N-k-1$   GO TO 4   $[\Lambda_{k,i-1} \in (\Lambda_k, \Lambda_{k+1})]$
      $s_k = N-k$     GO TO 5   $[\Lambda_{k,i-1} \in (\Lambda_{k-1}, \Lambda_k)]$
      $s_k < N-k-1$   GO TO 6   $[\Lambda_{k,i-1} \in (\Lambda_{k+1}, \infty)]$
      $s_k > N-k$     GO TO 7   $[\Lambda_{k,i-1} \in (-\infty, \Lambda_{k-1})]$
4.
  4.1. Let $b_0 = \Lambda_{k,i-1}$.
  4.2. Let $a_0 = b_0 - \delta$.
  4.3. Compute $s(a_0)$.
  4.4. If:
          $s(a_0) = N-k$ then STOP and recycle
                                   $(a_0 + b_0)/2$.
```

```
            $s(a_0) = N-k-1$ then $b_0 = a_0$ and GO TO 4.2.
            $s(a_0) > N-k$ then run Algorithm 6.4.
5.
   5.1. Let $a_0 = \Lambda_{k,i-1}$.
   5.2. Let $b_0 = a_0 + \delta$.
   5.3. Compute $s(b_0)$.
   5.4. If:
            $s(b_0) = N-k-1$ then STOP and recycle
                                        $(a_0 + b_0)/2$.
            $s(b_0) = N-k$ then $a_0 = b_0$ and GO TO 5.2.
            $s(b_0) < N-k-1$ then run Algorithm 6.4.
6.
   6.1. Let $b_0 = \Lambda_{k,i-1}$.
   6.2. Compute $a_0$ from (6.1.3).
   6.3. Run Algorithm 6.4.
7.
   7.1. Let $a_0 = \Lambda_{k,i-1}$.
   7.2. Compute $b_0$ from (6.1.3).
   7.3. Run Algorithm 6.4.
```

In the early iterations of Algorithm 5.6 we may find that Algorithm 6.4 is run quite often at Steps 4.4, 5.4, 6.3, or 7.3 of Algorithm 6.5. However, in later iterations when $\Lambda_{k,i-1}$ becomes a good approximation to Λ_k we find that at either Step 4.4 or 5.4 the algorithm stops the first time through. This signifies that $\Lambda_{k,i-1}$ is an adequate approximation to Λ_k.

Using this algorithm, we reran the example above. The results are shown in Table 6.4. The column headed "step #" shows the path taken through Algorithm 6.5 to obtain each value.

This example shows that the value computed at iteration i can indeed be a poor estimate of the desired eigenvalue. We observe how Algorithm 6.5 reacts to update that value so that convergence to the correct value is assured. It also shows that in higher iterations the recycled value can be an adequate estimate. In the example we see that from N = 44 to N = 54, $\Lambda_{k,i-1}$ is found to be within the proper interval.

Table 6.4

Recycling With Algorithm 6.5

i	N	Λ	$\bar{\lambda}$	step #
starting value		-.03721		
1	8	-.03890	-.03934	-
2	18	-.15867	-.13287	6
3	25	-.18042	-31.987	4
4	29	-.25238	-2.5246	5
5	40	-.06215	-.06437	7
6	44	-.06213	-.06222	5
7	48	-.06213	-.06214	5
8	49	-.06213	-.06214	4
9	51	-.06231	-.06232	4
10	53	-.06241	-.06241	4
11	54	-.06241	-.06241	4

Chapter 7

SINGULAR PROBLEMS

7.1 Introduction

The next topic we discuss is that of singular S-L problems. Some of the theory of the singular problem is presented in Chapter 2. In the present chapter we show the additional capacity of our adaptive algorithm to solve the two most common classes of singular problems.

We recall the Sturm-Liouville system written in Liouville normal form.

$$y'' + (\lambda - q(x))y = 0$$

$$y(a) = 0 \quad , y(b) = 0$$

For problems in the normal form we find that singular problems generally fall into the following two classes.

a. The interval (a,b) is infinite, i.e. $a = -\infty$ or $b = \infty$ or both. The boundary conditions of such a problem are of the form $y(x) \to 0$ as $x \to \infty$.

b. The function $q(x)$ is discontinuous at a or b or both.

Our adaptive method is equipped to handle these types of problems.

Both types of problem are solved by truncating the interval (a,b) and solving the regular S-L problem defined on the truncated interval. By restricting the class of problems as indicated in Section 6.1.2, i.e. $q \in C(a,b)$ and any continuous spectrum is greater than the point spectrum, we are assured that as $h \to 0$ and the truncated interval $\to (a,b)$, the eigenvalues and eigenfunctions of the truncated problem will converge to those of the singular problem.

7.2 Infinite Domain

Existing techniques for solving the singular Sturm-Liouville problem with an infinite domain generally lie in two categories. First is the technique of mapping the infinite interval onto a finite interval. For example, from Chapter 3 we recall that Bailey, Gordon, and Shampine[1] transform all problems, regular or singular, to the interval (-1,1).

The second technique is to restrict the domain to some finite interval, i.e. instead of (a,∞), we solve the problem on (a,R), imposing the boundary condition $y(R) = 0$, where R is some appropriately chosen large number. Since we have already assumed that all problems are transformed to normal form we do not wish to require even further transformation. Thus we choose the second technique as the basis for our method.

The question, and one which cannot be answered without prior knowledge of the solution, y(x), is how large to make R. If R is too small we may truncate a significant portion of the eigenfunction thus greatly affecting the eigenvalue. If R is too large we will have introduced extra mesh points which have no effect on the eigenvalue. This would be counter-productive to the goals of our adaptive method.

To be truly adaptive our algorithm must be capable of adjusting R. In general we find that the optimum value of R depends not only on the problem but on which eigenvalue is desired within a given problem. Fischer and Usmani[17] have investigated the infinite domain with regard to two-point boundary value problems. We use some of their ideas here.

We start by assuming that some value of R is available. Some knowledge of the problem may make a reasonable guess at R possible. If not, the algorithm uses some preassigned value. We also assume that $b = \infty$ while a is finite. All results are easily adapted to the case in which $a = -\infty$, and we show some numerical results for each case in Section 7.4.

When the matrix problem

$$(A - \Lambda B)Y = 0 \qquad (7.2.1)$$

is solved on the interval (a,R), we make the assumption that $Y_{N+1} = 0$, however, $y(x_{N+1}) \neq 0$. Thus the actual solution (λ, y) of the differential equation satisfies

$$(A - \lambda B)\bar{y} = \bar{\tau}(\bar{y}) - \bar{e}_R \qquad (7.2.2)$$

where $\bar{e}_R = (0,0,\ldots,\epsilon_R)^t$ with

$$\epsilon_R = \alpha_{2N} y(x_{N+1}) + \beta_{2N} y''(x_{N+1}) \neq 0.$$

This is seen by looking at the last equation of the system (7.2.2),

$$\alpha_{0N} y(x_{N-1}) + 2y(x_N) + \alpha_{2N} y(x_{N+1}) + \beta_{0N} y''(x_{N-1}) + \beta_{1N} y''(x_N) + \beta_{2N} y''(x_{N+1}) = \tau(x_N).$$

If we now repeat the derivation of equation (4.3.21) of Section 4.3 starting, however, with equation (7.2.2) rather than with (4.3.9) we find that

$$\Delta\lambda = - Y^t D^{-1} D^{-1} \{ \bar{\tau}(\bar{y}) - \bar{e}_R \} / Y^t D^{-1} D^{-1} B\bar{y}.$$

If we assume that $\Delta\lambda\Delta y$ is negligible we get

$$\Delta\lambda \approx - \frac{Y^t D^{-1} D^{-1} \bar{\tau}(\bar{y}) - Y^t D^{-1} D^{-1} \bar{e}_R}{Y^t D^{-1} D^{-1} BY}. \qquad (7.2.3)$$

Our D^{-1} solution-weighted strategy is based on the expression (4.3.22). We found that we should require that

$$|d_{ii}^{-2} Y_i \ T(x_i)| \approx |d_{jj}^{-2} Y_j \ T(x_j)|.$$

We now see from (7.2.3) that the size of $Y^t D^{-1} D^{-1} \bar{e}_R$ should also be monitored.

Since $\bar{e}_R$ contains zeros everywhere but the last row, we know that

$$Y^tD^{-1}D^{-1}\bar{e}_R = d_{NN}^{-2}\, Y_N\, \epsilon_R$$

and so our strategy is to find the value of R at which $|d_{NN}^{-2}\, Y_N\, \epsilon_R|$ is sufficiently small.

Of course ϵ_R is not a known quantity since it involves the exact solution, y(x). Therefore we must be able to approximate ϵ_R using the computed solution, Y.

We know that

$$\begin{aligned}\epsilon_R &= \alpha_{2N} y(x_{N+1}) + \beta_{2N} y''(x_{N+1}) \\ &= \{\alpha_{2N} + \beta_{2N}(q_{N+1} - \lambda)\} y(x_{N+1}). \qquad (7.2.4)\end{aligned}$$

The value of $y(x_{N+1})$ must be approximated in (7.2.4). We cannot use Y_{N+1} because it has been set to zero. The best value available to us is Y_N. We approximate λ by Λ. Thus in our adaptive algorithm we monitor the size of

$$\delta_R = |d_{NN}^{-2}\, Y_N^2\, \{\alpha_{2N} + \beta_{2N}(q_{N+1} - \Lambda)\}|.$$

To insure that δ_R is small enough so that no significant error is introduced we require that

$$\delta_R \approx |d_{jj}^{-2}\, Y_j\, T(x_j)|/100.$$

This factor of 1/100 is arbitrarily chosen and can be considered an area for possible tuning of the algorithm.

If δ_R is not small enough, a new point, x_{N+2}, is added to the mesh at

$$x_{N+2} = x_{N+1} + h_\infty.$$

We let h_∞ start with the value of h calculated in the first iteration of the algorithm. Each time h_∞ is used after that its size is doubled. This ensures that R gets large quickly enough so that extra points are not introduced and extra iterations not performed. This factor of 2 is again an area for possible tuning of the algorithm.

The algorithm we use for singular problems of the first type is as follows:

<u>ALGORITHM 7.1</u>: This algorithm is exactly the same as Algorithm 5.6 but we include the following statement in Step 5i).

```
If $|d_{NN}^{-2} Y_N^2 \{\alpha_{2N} + \beta_{2N}(q_{N+1} - \Lambda)\}| > EPS/100$ then:
    a. add a point $x_{N+2} = x_{N+1} + h_\infty$.
    b. $h_\infty \leftarrow 2h_\infty$
```

Results for problems A - D of Table 7.1 are examples of Algorithm 7.1 in operation.

7.3 <u>Singularity in q(x)</u>

When the function q possesses a singularity at a or b, we encounter a similar problem. We assume for most of this discussion that a is the singular point although all arguments are valid for b as well.

It is obvious that the point a itself cannot be an element of the mesh because the difference equation (4.3.1) would require q(a) to be evaluated. Therefore we approximate the problem by letting x_0 be located at some point close to a, but where $q(x_0)$ is defined.

Our adaptive algorithm must decide if x_0 is close enough to a to ensure that $\Delta\lambda$ is small, and if not it must introduce a new point between a and x_0.

The error caused by this truncation occurs in the first equation of (7.2.1). We again have

$$(A - \lambda B)\bar{y} = \bar{\tau}(\bar{y}) - \bar{e}_0$$

where now $\bar{e}_0 = (\epsilon_0, 0, \ldots, 0)^t$ with

$$\epsilon_0 = \alpha_{21} y(x_0) + \beta_{21} y''(x_0).$$

As in Section 7.2, we approximate ϵ_0 by $\{\alpha_{21} + \beta_{21}(q_0 - \Lambda)\}Y_1$ and we monitor

$$|d_{11}^{-2}\, Y_1^2\, \{\alpha_{21} + \beta_{21}(q_0 - \Lambda)\}|.$$

The algorithm we use to accomplish this task is:

<u>ALGORITHM 7.2</u>: This algorithm is also like Algorithm 5.6 but with the following additions.

1. In Step 1 the basic mesh is set up on $[x_0, b]$. The user can supply an initial value for x_0, otherwise $x_0 = a + (b-a)/8$.
2. In Step 5 include:

   If $|d_{11}^{-2}\, Y_1^2\{\alpha_{21} + \beta_{21}(q_0 - \Lambda)\}| > EPS/100$
   then split x_0. (add point at $(a+x_0)/2$)

Problems C and D of Table 7.1 are of the second type.

The inclusion of ϵ_0 into our algorithm does not capture all of the error introduced by the truncation. Error is propagated throughout the entire eigenfunction by this truncation process. Techniques for estimating this error may lead to even better error estimates than those shown in Table 7.1.

7.4 Numerical Results

In this section we show results of running Algorithms 7.1 and 7.2 on several singular problems. We solve the following problems:

Problem A: (Morse potential problem)

$$y'' + (\lambda - q(r))y = 0$$

$$y(0) = 0, \; y(r) \to 0 \text{ as } r \to \infty$$

where

$$q(r) = D[1 - \exp(-a(r - r_e))]^2 - D$$

with

$$D = 188.4355, \; a = 0.711248, \; r_e = 1.9975.$$

$$\lambda_0 \approx -178.79850, \; \lambda_4 \approx -110.80832.$$

Problem B:

$$y'' + (\lambda - x^2)y = 0$$

$$y(-\infty) = y(\infty) = 0$$

$$\lambda_n = 2n + 1, \; n=0,1,2,\ldots$$

$$y_n(x) = \exp(-x^2/2)H_n(x)$$

where $H_n(x)$ is the Hermite polynomial defined by: $H_0(x) = 1$,

$H_1(x) = 2x$, $H_{k+1}(x) = 2xH_k(x) - 2kH_{k-1}(x)$, $k=0,1,\ldots$

<u>Problem C</u>:

$$y'' + (\lambda + \frac{1}{x})y = 0$$

$$y(0) = 0, \; y(x) \to 0 \text{ as } x \to \infty.$$

$$\lambda_n = -1/4(n+1)^2, \quad n=0,1,2,\ldots$$

<u>Problem D</u>:

$$y'' + (\lambda + \frac{1}{x} - \frac{L(L+1)}{x^2})y = 0$$

$$y(0) = y(\infty) = 0$$

$$\lambda_n = -1/4(L+n+1)^2, \; n=0,1,2,\ldots$$

Table 7.1 shows, for each of the above problems, the same information provided by Table 5.6. In addition we present the calculated values of a and b.

The increased accuracy obtained by the adaptive method is quite evident in the results of Table 7.1. The values of a and b, used to compute the Numerov results, are those calculated by the adaptive method. Normally, however, the user of a uniform mesh method must select his own a and b. If some other standard values are chosen for the problems above the improved accuracy of the adaptive method becomes even more apparent.

Table 7.1

Singular Problems

Mth	TOL	N	a/b	$\lvert e\rvert$	$\lvert e_{est}\rvert$	$\lvert e_c\rvert$
Problem A,λ_0						
A	10^{-4}	41	0	1.114D-4	6.083D-5	5.053D-5
			3.375			
N		41		2.872D-4	3.189D-4	3.169D-5
		64		4.058D-5	5.436D-5	1.378D-5
Problem A,λ_4						
A	10^{-4}	270	0	1.918D-4	3.695D-5	1.548D-4
			14.625			
N		270		4.874D-3	4.495D-3	3.792D-4
Problem B,λ_0						
A	10^{-6}	136	-8.75	7.450D-7	6.808D-7	6.419D-8
			8.75			
N		136		2.147D-6	2.132D-6	1.526D-8
		256		1.707D-7	1.704D-7	3.432D-10
Problem B,λ_2						
A	10^{-4}	70	-16.75	1.133D-4	8.673D-5	2.658D-5
			16.75			
N		70		1.084D-2	9.222D-3	1.618D-3
		128		9.311D-4	8.872D-4	4.393D-5
Problem C,λ_0						
A	10^{-5}	80	1.097D-7	2.702D-6	2.530D-6	1.727D-7
			27.48125			
N		80		4.047D-3	6.899D-4	3.357D-3
		128		1.706D-3	3.094D-4	1.396D-3

Table 7.1 (continued)

Mth	TOL	N	a/b	$\lvert e \rvert$	$\lvert e_{est} \rvert$	$\lvert e_c \rvert$
Problem C,λ_2						
A	10^{-5}	92	1.097D-7	5.513D-7	5.327D-7	1.849D-8
			177.10625			
N		92		2.252D-3	2.032D-4	2.049D-3
		128		1.454D-3	1.604D-4	1.293D-3
Problem D,λ_0,L=3						
A	10^{-5}	40	.025	4.114D-6	3.326D-6	7.880D-7
			336.7063			
N		40		9.935D-6	1.184D-5	1.910D-6
		64		1.880D-6	1.470D-5	4.095D-7
Problem D,λ_2,L=3						
A	10^{-5}	86	.0125	1.506D-6	1.470D-6	3.647D-8
			655.5906			
N		86		8.446D-6	7.949D-6	4.971D-7
		128		1.803D-6	1.492D-6	3.110D-7

The diversity of the calculated a's and b's shows that it would be difficult to select these numbers beforehand. A wide range of values for h_i is also characteristic of these problems. For example, in Problem C, h_i takes on values from 1.907×10^{-7} to 4.9875 when computing λ_2. Similar ranges are found in all the problems of this chapter. No user can be expected to choose such a mesh in advance.

In Figures 7.1 - 7.4 we show the point placement for Problems B and C. In these graphs we note the diversity of the h_i's as well as the dense mesh in steep sections of the functions and sparse mesh in level sections.

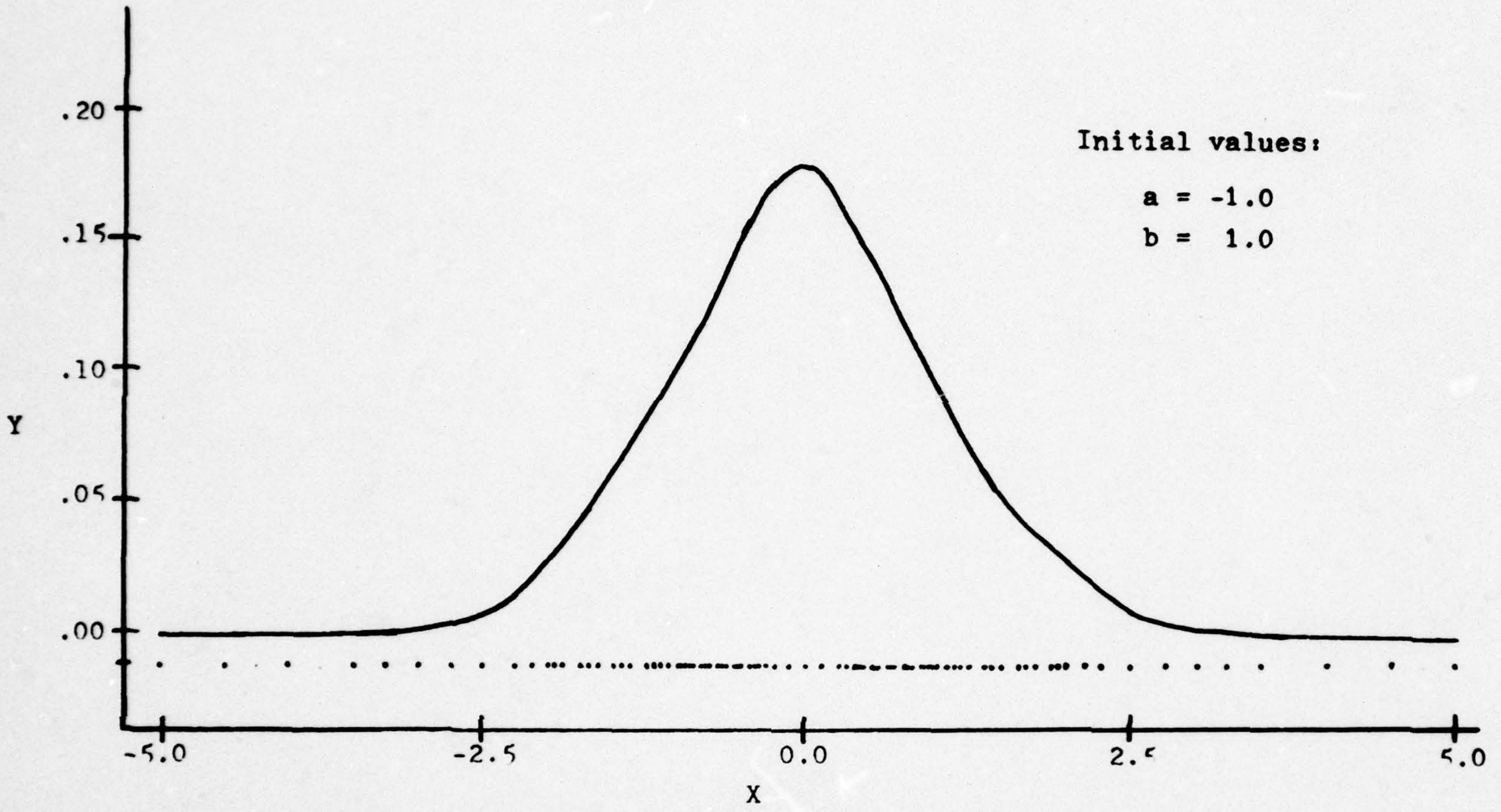


Figure 7.1

Mesh Distribution for Problem B, λ_0

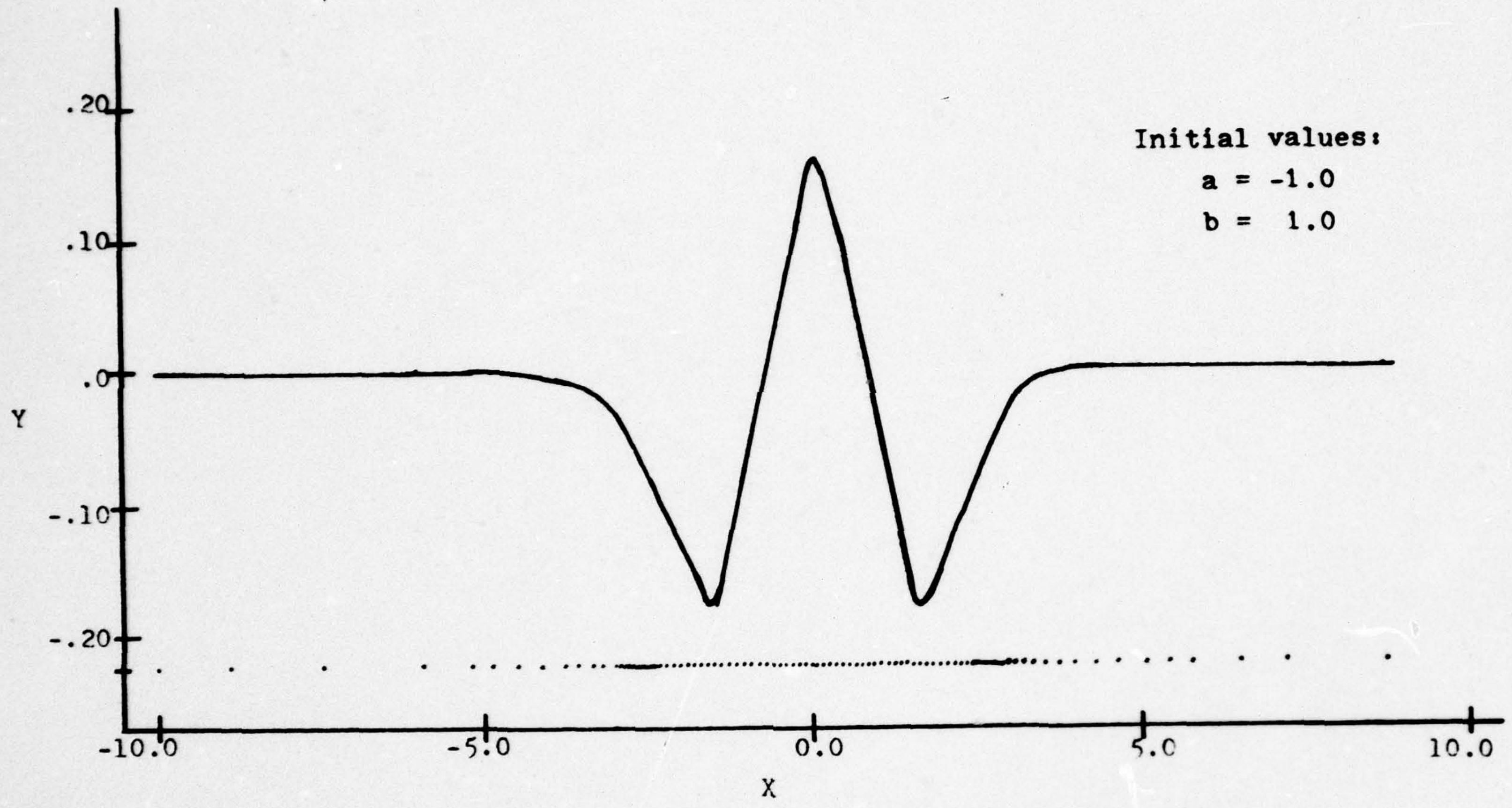


Figure 7.2

Mesh Distribution for Problem B, λ_2

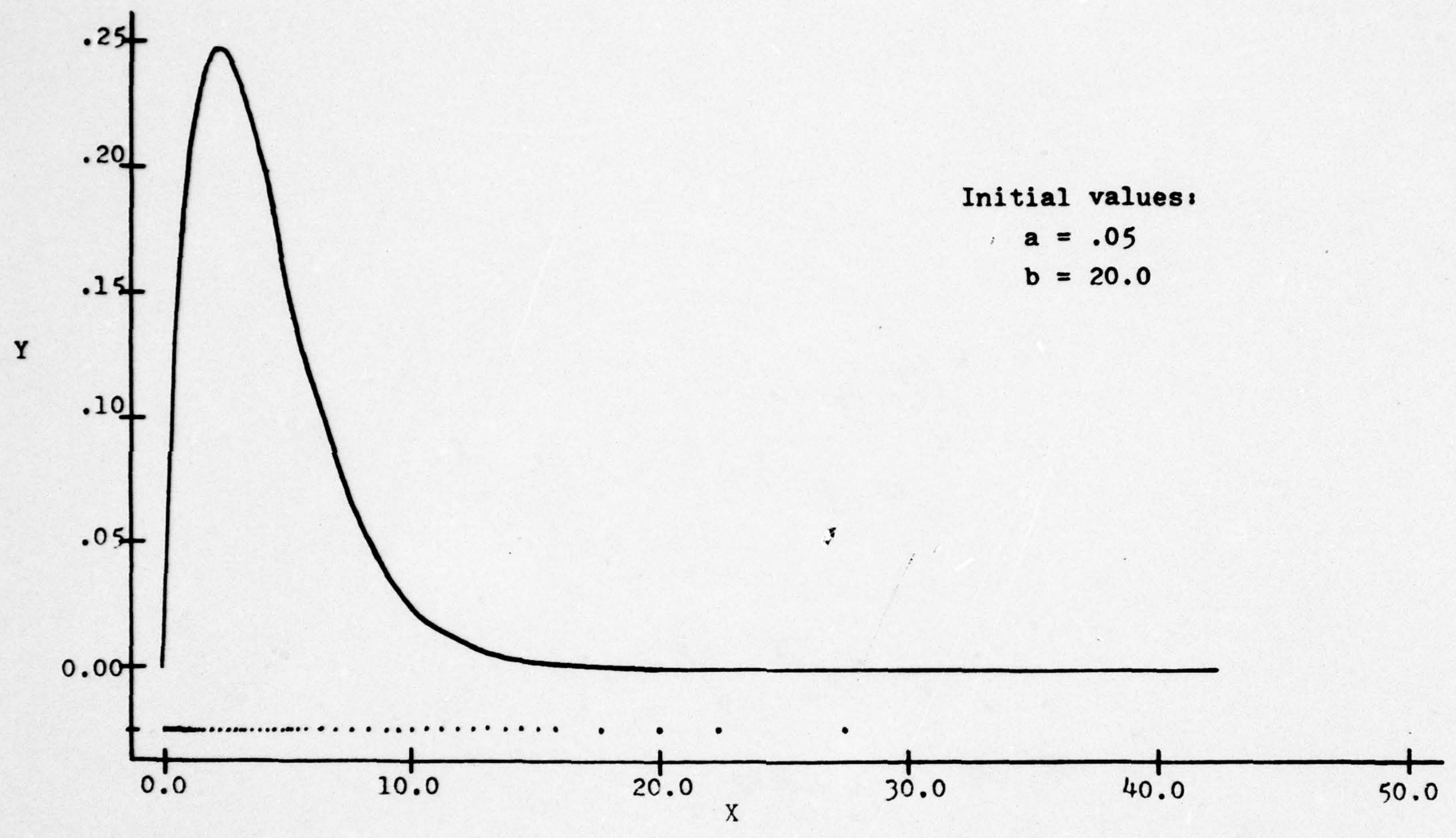


Figure 7.3

Mesh Distribution for Problem C, λ_0

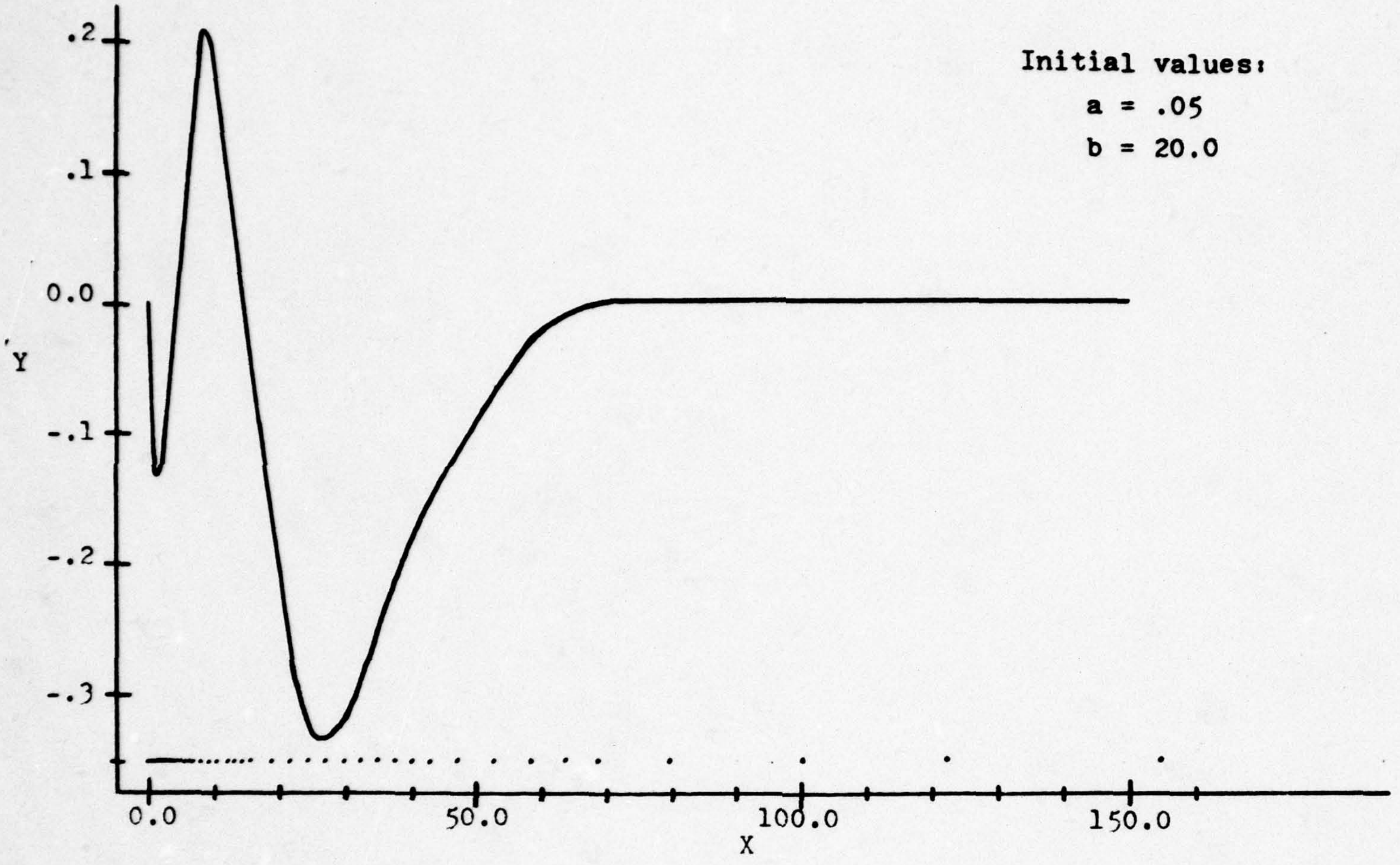


Figure 7.4

Mesh Distribution for Problem C, λ_2

7.5 Order of Convergence

When the problem to be solved has a singularity the order of convergence results of Section 4.3 are no longer necessarily valid.

Let us consider, for example, Problem C. The adaptive method solves the problem by truncating the interval $(0,\infty)$ to (x_0,R). We now solve a perturbed problem which is regular on (x_0,R). We would obtain $O(h^4)$ convergence of the matrix problem to the perturbed problem as $h \to 0$, however, convergence of the perturbed problem to the original singular S-L problem, as $h \to 0$, $x_0 \to 0$, and $R \to \infty$, must be analyzed separately for each problem.

The convergence rate will now depend on $h_0 = (x_0 - a)$ and on R. In this example we find that while solving for λ_0 we have $|e|_8 = 2.107\text{x}10^{-1}$ and $|e_c|_8 = 2.095\text{x}10^{-1}$ with $h_0 = .05$. After convergence we have $|e|_{80} = 2.702\text{x}10^{-6}$ and $|e_c|_{80} = 1.727\text{x}10^{-7}$ with $h_0 = 1.907\text{x}10^{-7}$. Computing α and β as we did in Section 5.3.4, we find that $\alpha \approx .903$ and $\beta \approx 1.123$. These figures imply that

$$(\Lambda - \lambda) \approx O(h)$$

and $$(\Lambda - \bar{\lambda}) \approx O(h).$$

In general, these values of α and β will be different for each singular problem.

7.6 Calculation of a Transformation Function

Many methods for solving the singular S-L problem involve a mapping of the problem onto some other domain. The independent variable, x, is mapped to a new variable r by the transformation

$$r = f(x).$$

Examples of such transformations include exponential functions ($r = e^{-kx}$), log functions ($r = \log x$), and others (e.g. $r = \frac{1}{1+x}$).

The purpose of such a transformation is to alter the differential equation so that a uniform mesh method can be used effectively. As an example we look at the hydrogen atom equation, Problem C of Chapter 7. We know that a solution can only be found efficiently if the mesh is fine near zero and very coarse for larger x. The transformation f(x) should thus spread out the smaller values while compressing larger ones. The adaptive method of this thesis is useful in selecting such a transformation.

We want a function f(x) with the property that

$$\Delta r = f'(x)\Delta x \approx \text{constant}.$$

We solve the example problem using the adaptive algorithm. This gives us an optimal mesh of Δx's. We now look for a function which satisfies

$$f'(x)\, \Delta x \approx \text{constant}.$$

For this problem we found

$$f(x) = \frac{1}{1+x}$$

to be an appropriate function. Table 7.7 shows values of $f'(x)\Delta x$ for every x at which a step size change occurs.

N was computed to be 67 for this problem. Thus the transformed problem should use $\Delta r = \frac{1}{67} \approx 1.4925D\text{-}2$. We see that many of the values in Table 7.2 are close to this number. Discrepancies are due to the discontinuity of our computed mesh.

Such analysis would need to be done for each problem and selection of a proper f(x) could be difficult. The results of this section show, however, that such a use of the adaptive method can be worthwhile for selecting a transformation if a uniform mesh method must be used.

Table 7.2

$f'(x)\Delta x$ for Problem C

x	$f'(x)\Delta x$
1.250D-2	1.219D-2
5.000D-2	3.534D-2
1.279D-1	1.531D-2
2.838D-1	1.494D-2
6.734D-1	5.566D-2
1.609D0	4.581D-2
5.661D0	1.405D-2
1.626D1	4.186D-3
2.249D1	4.518D-3

Chapter 8

PERFORMANCE EVALUATION

In this chapter we evaluate the performance of the adaptive algorithm in terms of space and time utilization. We show relationships between the index of the desired eigenvalue and the space and time requirements.

8.1 Space Requirements

The measure of space requirements of our adaptive algorithm is N, the number of points in the final mesh.

We know from the many problems of Chapters 5 and 7 that for a given problem N is a function of both i, the index of the desired eigenvalue, and TOL, the allowable error in the numerical solution. In order to simplify the analysis we assume that TOL is constant and we exhibit the behavior of N as i increases. Thus we define the function $S(i) = N$.

Figures 8.1 - 8.4 show the computed value of N plotted against i for several different problems. TOL is held constant at 10^{-2}.

8.1.1 Regular Problems

The function S is different for each problem and so a separate analysis must be performed on each to obtain a closed expression for S(i). Figures 8.1 - 8.3 suggest, however, that there is a pattern to the behavior of regular problems.

We look at the simplest of these problems to analyze, Problem I. We know from Chapter 4, (4.3.22), that when our adaptive method is used to solve this problem for λ_i we have

$$\Delta\lambda_i \approx - Y^t D^{-1} D^{-1} \bar{\tau}(\bar{y}) \,/\, Y^t D^{-1} D^{-1} BY.$$

This is the quantity we want to be constant for all i.

We let $G = 1 / Y^t D^{-1} D^{-1} BY$ and recall from Chapter 4 that $G = O(h^{-1})$. So we have

$$\begin{aligned} \Delta\lambda_i &\approx - G Y^t D^{-1} D^{-1} \bar{\tau}(\bar{y}) \\ &= - G \sum_{j=0}^{N+1} Y_j d_{jj}^{-2} \tau(x_j). \end{aligned}$$

Now, because we have assumed that the mesh is uniform almost everywhere, we know that

$$\tau(x_j) = O(h^6) y^{(vi)}(x_j) + O(h^8).$$

We recall Problem I to be

$$\begin{aligned} &y'' + \lambda y = 0 \qquad (8.1.1) \\ &y(0) = y(1) = 0 \\ &\lambda_i = (i+1)^2 \pi^2 , \quad i=0,1,2,\ldots \end{aligned}$$

Differentiating (8.1.1) 4 times we get

$$y^{(vi)}(x) = - \lambda^3 y(x)$$

and so we have

$$\Delta\lambda_i \approx G \sum_{j=0}^{N+1} Y_j d_{jj}^{-2}\, O(h^6) \lambda_i^3 y(x_j).$$

Since we know the eigenvalues of the problem, we can replace λ_i^3 by $(i+1)^6\pi^6$ or simply $O(i^6)$. Thus

$$\Delta\lambda_i \approx G \sum_{j=0}^{N+1} Y_j d_{jj}^{-2}\, O(h^6)\, O(i^6)\, y(x_j). \qquad (8.1.2)$$

But (8.1.2) is a sum of N + 2 or O(N) terms so we have

$$\Delta\lambda_i \approx G\, O(N)\, O(h^6)\, O(i^6).$$

Now replacing G by $O(h^{-1})$ and $O(h^6)$ by $O(N^{-6})$ {because $h = O(1/N)$} we have

$$\Delta\lambda_i \approx O(i^6)\, O(N^{-4}).$$

This says that for $\Delta\lambda_i$ to be constant for all i, N must increase like $i^{3/2}$. The solid lines on Figures 8.1 - 8.3 show functions of the form

$$N = S(i) = c_0 + c_1\, i^{3/2} \qquad (8.1.3)$$

fitted to the data by least squares.

Actually each regular problem must be analyzed individually to get such a relationship. However, we note that since eigenfunctions of all regular problems are similar in shape we might expect the S(i) to be similar also. This hypothesis is given even more credence by observing the excellent fits of the least squares functions in Figures 8.2 and 8.3.

The exponent, 3/2, in equation (8.1.3) is a result of several approximations or assumptions. Because of this, we cannot place too much faith in the exact value of 3/2. We have, however, fitted linear and quadratic least squares functions to the data of Figure 8.1. When we compare the values

$$\sum_{i=0}^{9} (N - S(i))^2$$

for the 3 different functions, S(i), we find that the value for (8.1.3) is smallest. (3709.0 compared to 3710.9 and 5087.9). Thus, while 3/2 may not be the exact exponent, it does appear that the value lies between 1 and 2.

8.1.2 Singular Problems

The process used above was also performed on a singular problem, Problem B.

$$y'' + (\lambda - x^2)y = 0$$

$$y(-\infty) = y(\infty) = 0$$

$$\lambda_n = 2n + 1 , \quad n=0,1,2,\ldots$$

The problem was solved for λ_i, $i=0,1,\ldots,6$ using Algorithm 7.1, holding TOL constant at 10^{-2}. Figure 8.4 shows the plotted results. The data appears to be linear with respect to i. The solid line is a least squares straight line. The computed values of a and b range from (-3.5,3.5) to (-4.75,4.75). Each was started at (-1.0,1.0).

8.1.3 Conclusions of Space Requirement Findings

We found, for Problem I and possibly many other regular problems, that while space does increase exponentially with respect to i, it is not at a prohibitive rate.

Using the least squares function $N = S(i) = 18.136 + 7.921\, i^{3/2}$ we find, for example, that λ_{20} of Problem I could be computed with approximately 727 mesh points to an absolute accuracy of 10^{-2}. Since $\lambda_{20} \approx 4352.5$ this represents a relative accuracy of about 2.3×10^{-6}. In the singular problem we observed a linear relationship between N and i. In that problem λ_{20} could be computed with about 327 points to a relative accuracy of 2.4×10^{-4}.

We note, however, that the number of points required for a singular problem changes when the initial a and b are changed. If the final a and b were known before running the algorithm they could be input thus reducing the size of the mesh.

8.2 Time Requirements

The second measure of the performance of our method is time. Figures 8.5 - 8.8 show the actual CPU time required to compute each eigenvalue. Our test program is written in FORTRAN and compiled with the FORTRAN G compiler on an IBM 370/168. The time was measured by calling system clock routines at the start and at the end of the computation. The times shown are only for the portion of the program corresponding to the adaptive algorithm. Initialization and input/output are excluded. All times are in milliseconds. Exact values of the times are not what we wish to show but rather the relative times as i increases.

All the processes done in the adaptive algorithm are known to be linear in time with respect to N. This includes matrix reduction, mesh refinement, and coefficient and eigenvalue computation. Thus if times were measured for only the last iteration of each data set we would see N and T (time) related linearly while i and T would again be related like $T = O(i^{3/2})$. However, for the algorithm to arrive at the higher values of N it must compute eigenvalues for several lower values of N first. Each of these computations in turn takes several Rayleigh quotients iterations. Thus as i increases, thereby increasing N, the time shown is a sum of several iterations.

To analyze this relationship we define a new variable M to be the sum of N x (number of Rayleigh quotient iterations) summed over each N used to compute λ. For example, in computing λ_5 of Problem I we find that matrix problems of sizes N = 8, 16, 32, 38, 50, 52, 66, 78, 84, and 139 must be solved and that respectively 4, 4, 3, 4, 4, 4, 4, 3, 3, and 3 Rayleigh quotient iterations are required. Thus M = 1887.

If the process is indeed linear with respect to time within each iteration then we must expect that M and T are linearly related. Figure 8.9 shows M and T plotted for Problem I. The solid line is the least squares line T = 1.580 M + 73.940. Similar linear relationships exist for the other example problems of this chapter.

All the data in Figures 8.5 - 8.9 were obtained by initially letting N = 8. This value is, however, an input to the algorithm. If the user wants to decrease the time for higher eigenvalues he can set N = 16, 32,... or some higher value based on his knowledge of the problem, This would eliminate some of the earlier iterations thus reducing the time required for convergence. We must note, however, that these early iterations use much less time than the later ones because N is still small. Thus the time savings would possibly not be a considerable one.

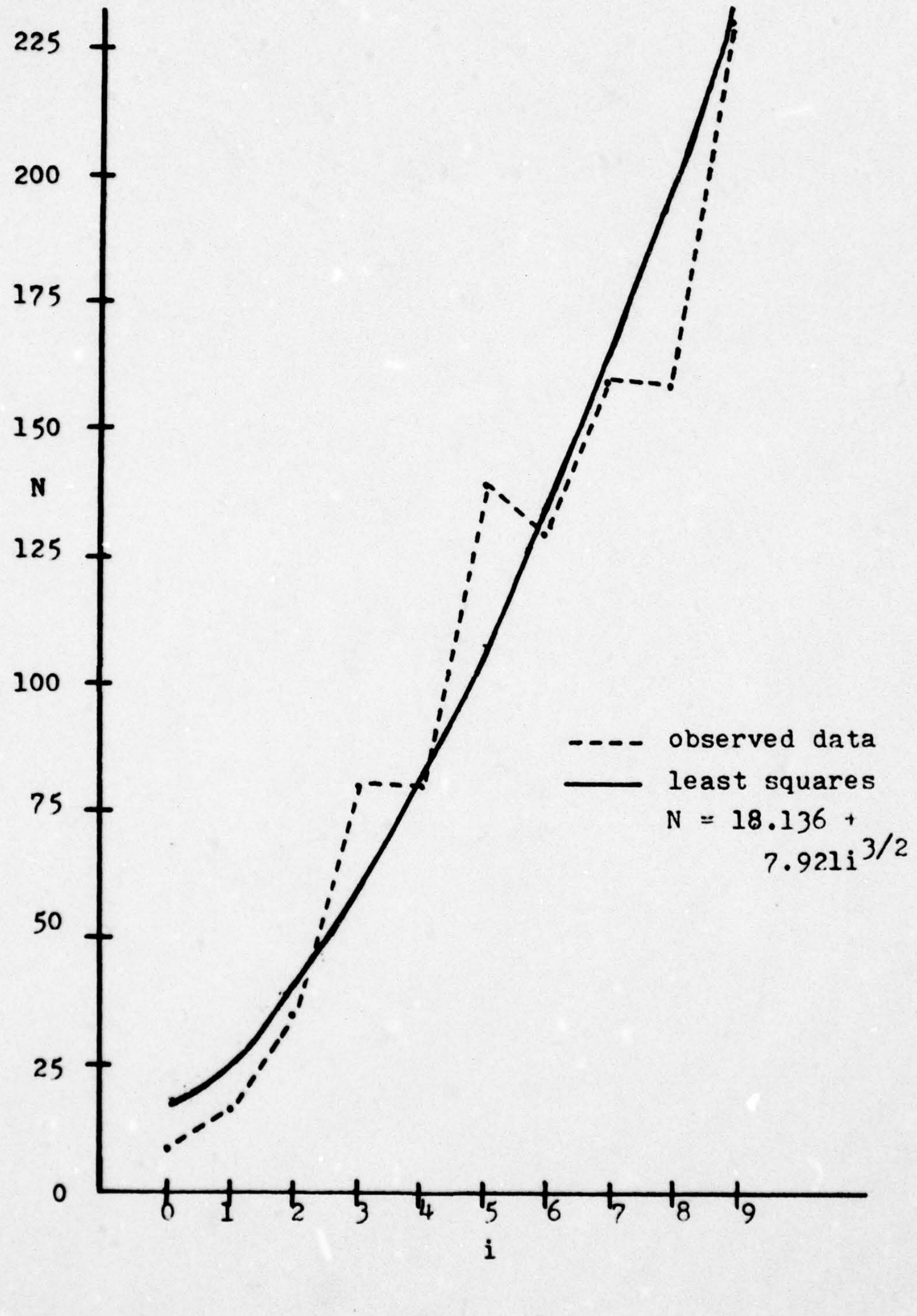


Figure 8.1

N vs i - Problem I

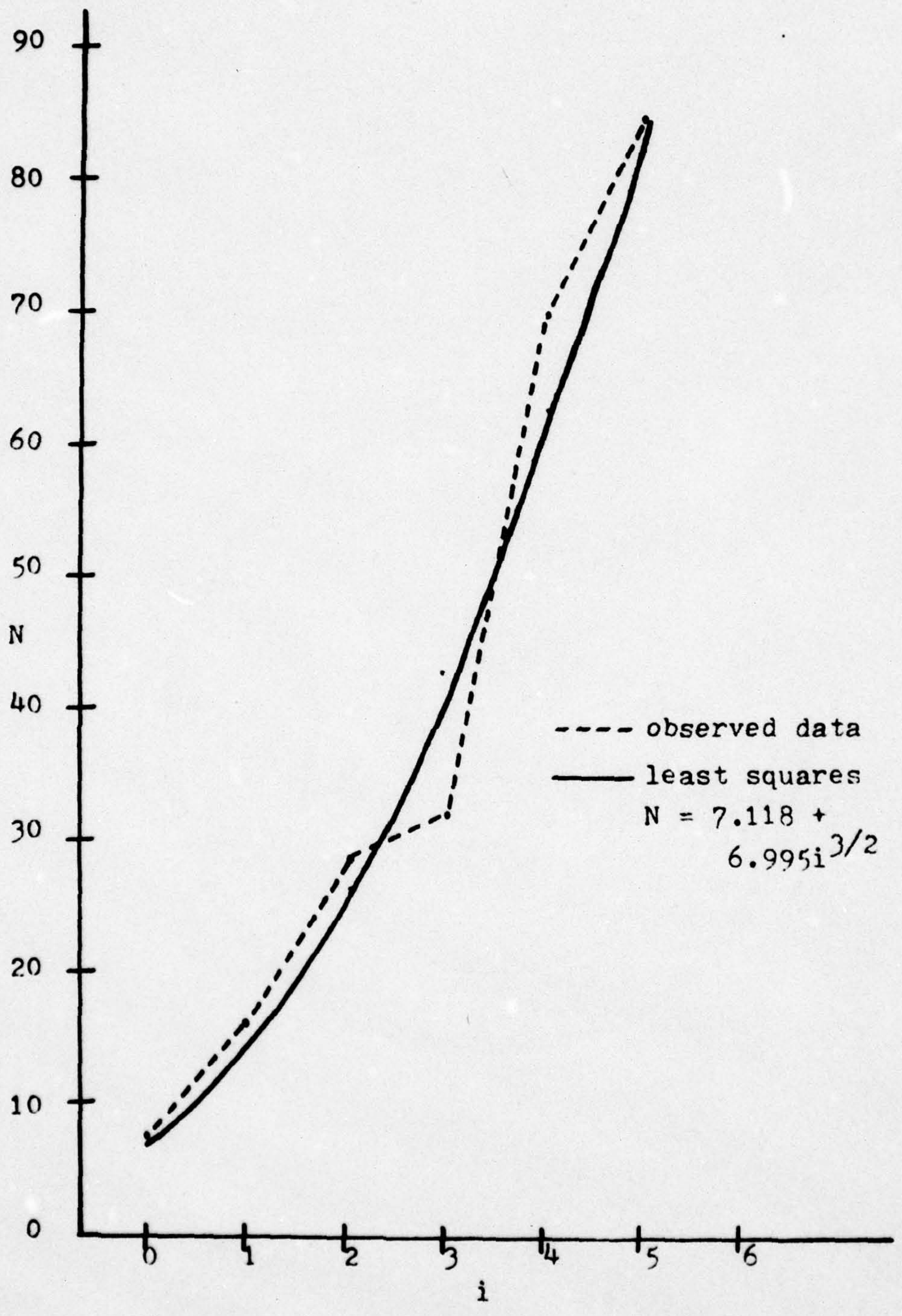


Figure 8.2

N vs i - Problem II

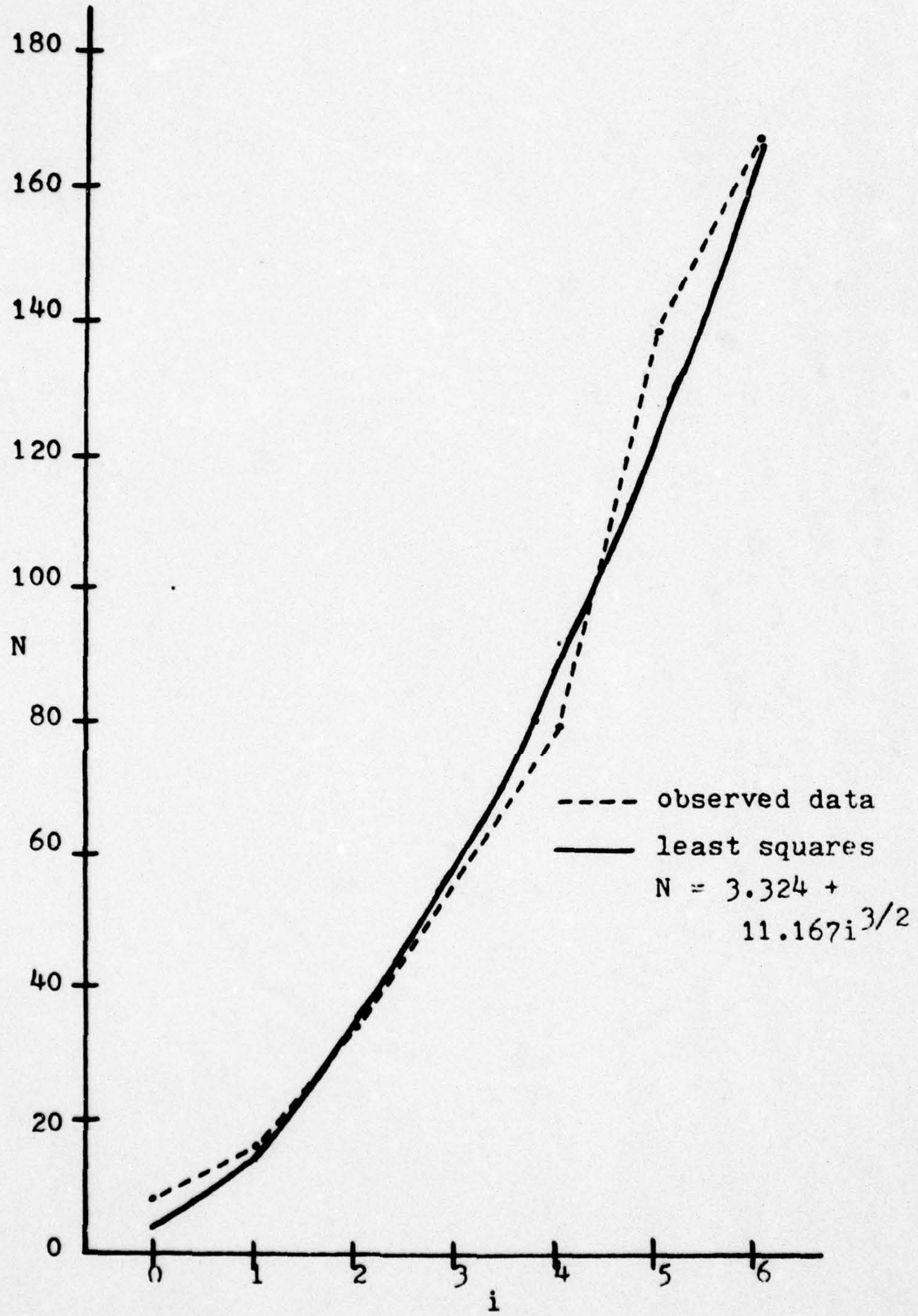


Figure 8.3

N vs i - Problem III

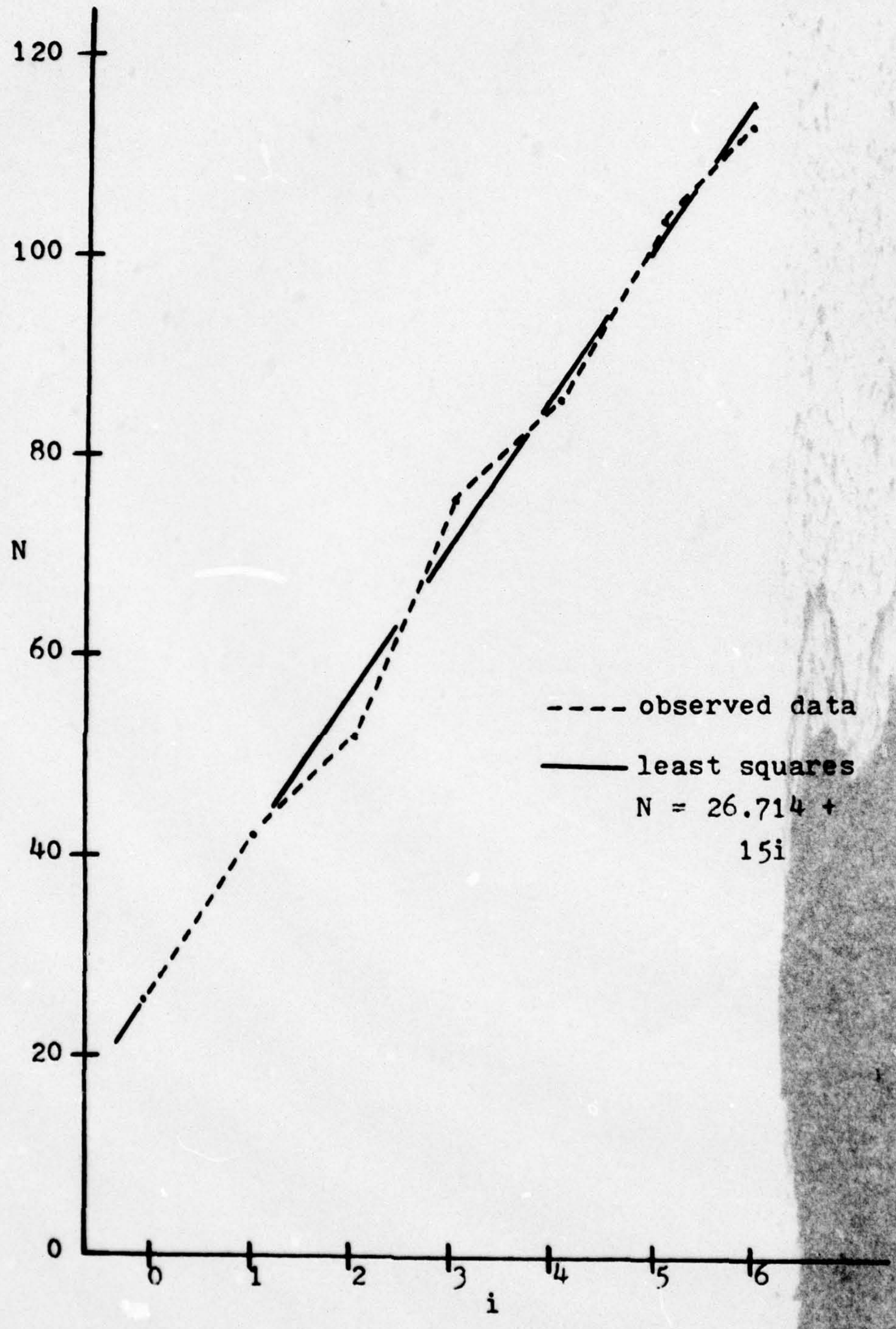


Figure 8.4

N vs i - Problem B

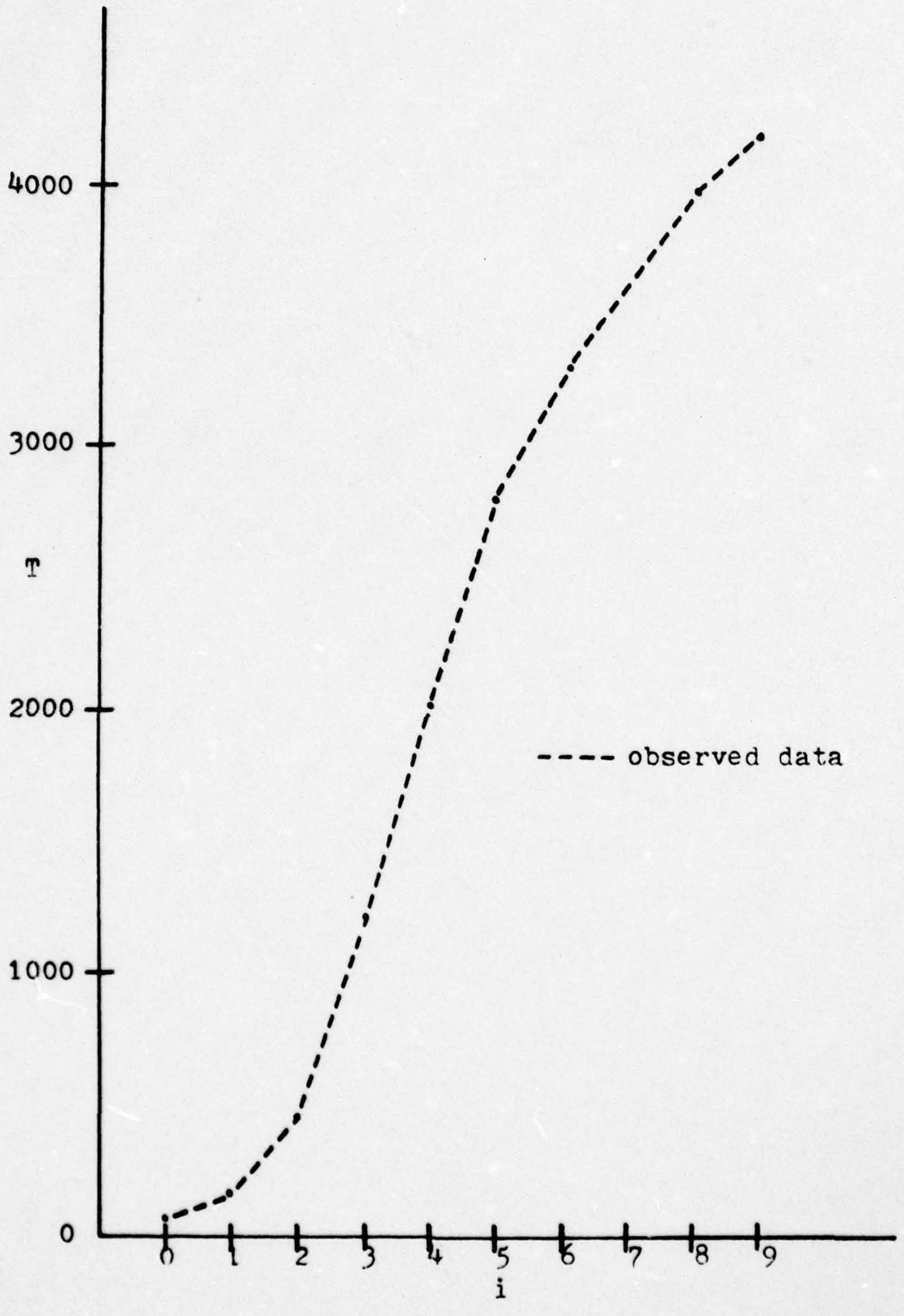


Figure 8.5

T vs i - Problem I

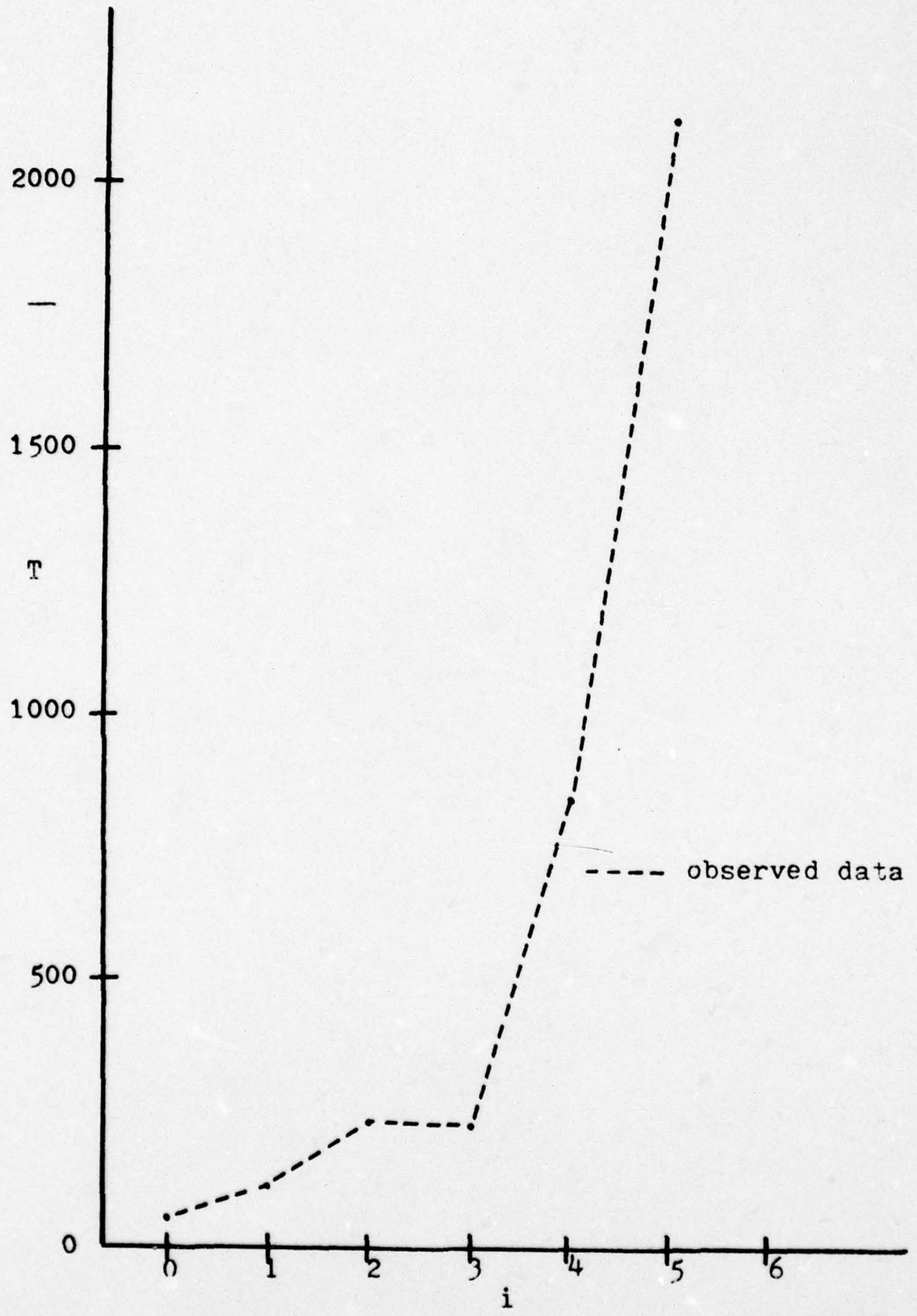


Figure 8.6

T vs i - Problem II

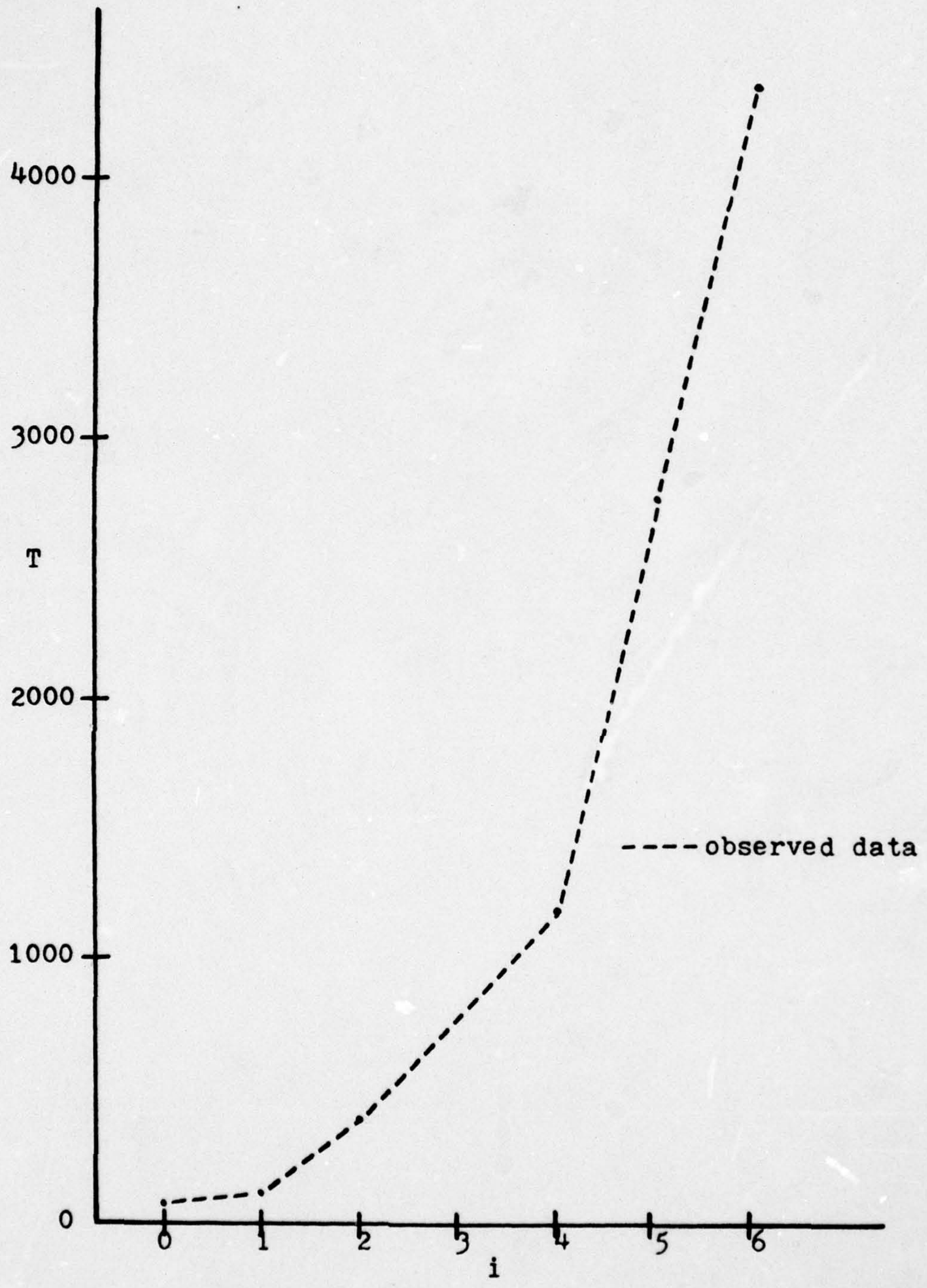


Figure 8.7

T vs i - Problem III

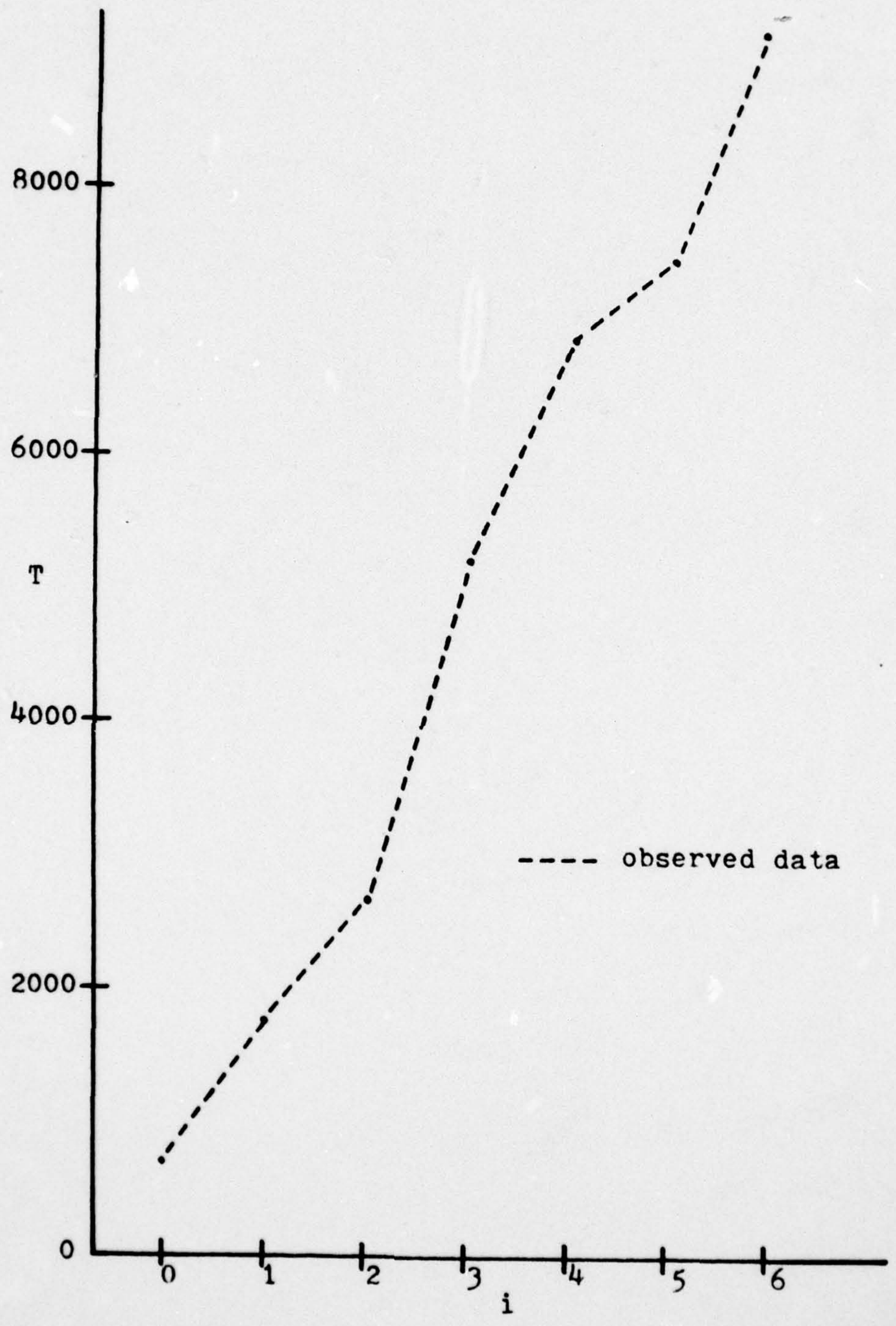


Figure 8.8

T vs i - Problem B

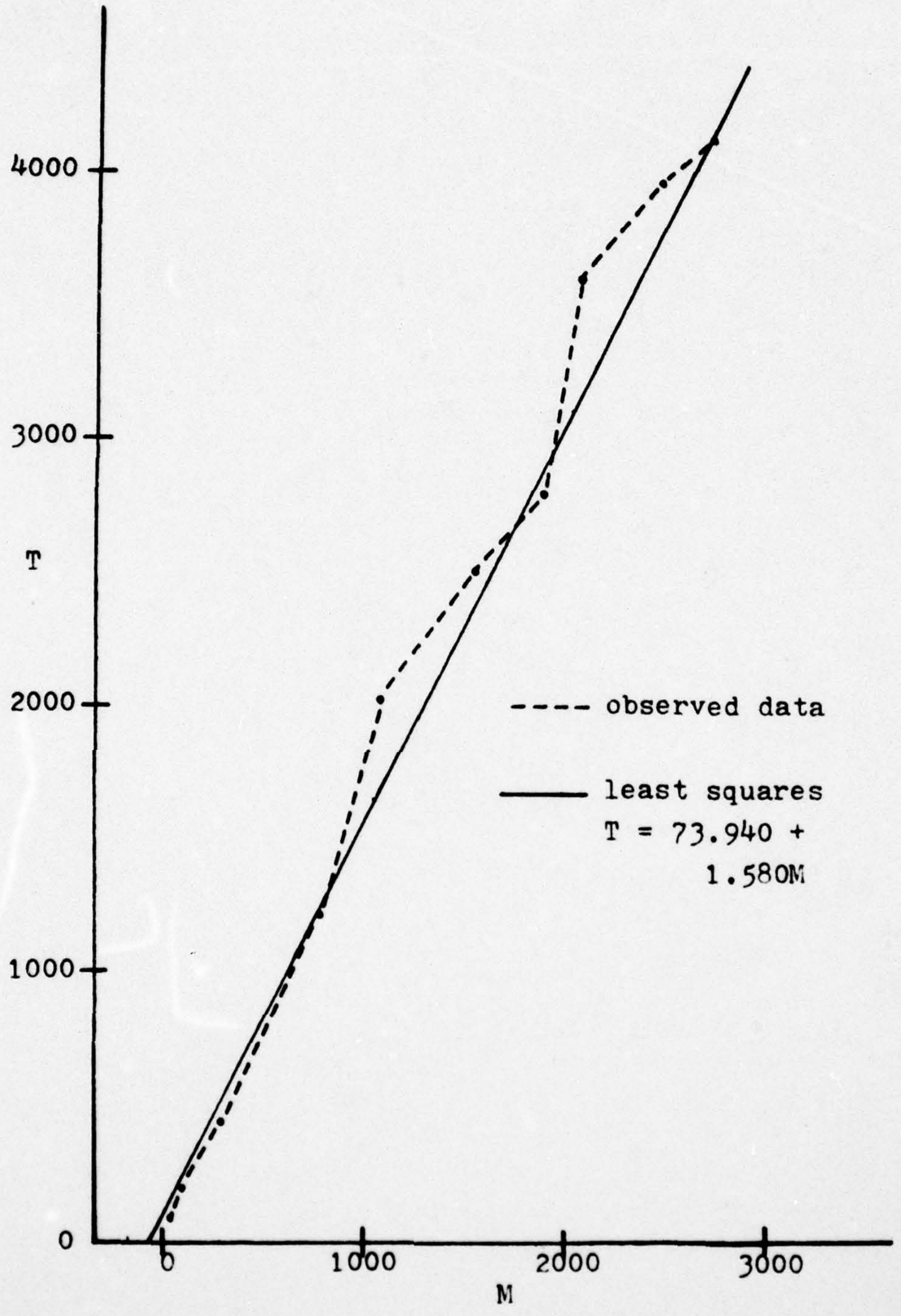


Figure 8.9

T vs M - Problem I

8.3 Comparison With an Initial Value Method

In this section we present some numerical results comparing our algorithm with SLEIGN, the software package described in [1] and briefly discussed in Section 3.5 of this thesis.

We compared performance on regular problems by running both algorithms on several sample problems. Comparative times (T), measured in milliseconds, are shown in Table 8.1 along with actual errors ($|e|$) and error estimates ($|e_{est}|$). Another important measure of performance is the value $|e_{est}|/|e|$, the ratio of error estimate to actual error. Ideally this ratio should be equal to 1. Any value significantly less than 1 denotes a small estimate which would have caused the algorithms to stop prematurely. A value greater than 1 indicates that the algorithm did extra work by making the actual error too small.

Data from SLEIGN for comparing performance on singular problems was taken directly from [1]. Times are not available so we look only at $|e_{est}|/|e|$. Problems B, C, and D were run by each algorithm and Table 8.2 shows the comparison. In addition, Table 8.3 shows values for all other eigenvalues computed.

TOL was set at 10^{-5} for problem I. For all other problems TOL was set at 10^{-4}.

Table 8.1

Adaptive vs SLEIGN - Regular Problems

Prob	Meth	N	T	$\lvert e \rvert$	$\lvert e_{est} \rvert$	Ratio	$\lvert e_c \rvert$
I, λ_0	A	44	1734	2.395D-6	2.393D-6	.9992	1.732D-9
	S	-	1767	3.052D-5	9.998D-6	.3276	-
II, λ_0	A	16	150	1.431D-5	1.350D-5	.9434	8.127D-7
	S	-	844	2.861D-6	6.337D-6	2.215	-
II, λ_2	A	68	894	3.129D-5	3.114D-5	.9952	1.533D-7
	S	-	5486	1.526D-5	2.827D-5	1.853	-
III, λ_0	A	20	306	4.062D-5	4.051D-5	.9973	1.179D-7
	S	-	1274	7.403D-5	2.520D-5	.3404	-

Table 8.2

Adaptive vs SLEIGN - Singular Problems
Computed by Both Algorithms

Problem	$\lvert e_{est} \rvert / \lvert e \rvert$ - Adaptive	$\lvert e_{set} \rvert / \lvert e \rvert$ - SLEIGN
B, λ_0	.9138	.3100
C, λ_0	.9396	10.00
D, λ_0	.8085	.2375

Table 8.3

Adaptive vs SLEIGN - All Singular Problems

Adaptive			SLEIGN		
Prob	Ratio(λ_0)	Ratio(λ_2)	Prob	Ratio(λ_0)	Ratio(λ_{10})
A	.5461	.1926	1	.3100	.7988
B	.9138	.7655	2	.2375	5.02
C	.9363	.9663	3	.1850	-
D	.8085	.9761	4	162.79	69.09
			5	6.895D-3	1.063D-2
			6	1.200	186.49
			13	10.00	1.135

(Note: Values are shown in Table 8.3 only for those problems for which the actual error is known to be reliable.)

This limited comparison appears to show that the adaptive method is preferable to this initial value method for two reasons. First, the error estimate obtained by our method is much more reliable than the one of the initial value method. Values of $|e_{est}|/|e|$ are consistantly around 1.0 for our method, ranging from .1926 to .9992, while the ratios for SLEIGN vary from 6.895×10^{-3} to 162.791. Second, the method appears to be significantly faster for similar accuracies. It must be noted, however, that SLEIGN solves the problem without requiring it to be put in the normal form.

Chapter 9

SUMMARY AND CONCLUSIONS

In this thesis we developed an adaptive finite-difference method for solving numerically the Sturm-Liouville problem. The Sturm-Liouville theory is presented in Chapter 2. Chapter 3 serves as background on the many other techniques available for solving this problem. We found that most current work is done with some variation of the initial value or shooting method. These methods, however, traditionally use Newton's method to find a root or eigenvalue. This leads to quadratic convergence to the desired eigenvalue.

In Chapter 4 we presented a non-uniform finite-difference method. We showed that when h, the maximum step size, is sufficiently small this method is capable of computing all eigenvalues and that cubic convergence is experienced via Rayleigh quotient iteration. We showed also that with one application of the deferred correction algorithm, the method is $O(h^6)$ in the eigenvalue and eigenvector.

Two algorithms were presented in Chapter 5. The first is a non-adaptive, iterative algorithm. It shows the capability of a non-adaptive method which uses a uniform mesh.

The second algorithm is our adaptive, iterative method. It uses the non-uniform finite-difference method of Chapter 4. The D^{-1} Solution-Weighted strategy is found to be the best for our problem and it is used in the algorithm. We found that the adaptive, iterative algorithm does indeed obtain accuracy equivalent to the uniform mesh method while using fewer grid points.

In Chapter 6 we solve the problem of finding a starting value for the method and for monitoring the recycled eigenvalue from iteration to iteration. We presented an efficient method for estimating the desired eigenvalue close enough to ensure proper convergence. Sturm sequences are used to accomplish these tasks. Thus, our method is self-starting, making it more practical to use than most other current methods.

Singular problems were addressed in Chapter 7. We showed that with a few additions to the adaptive algorithm, singular problems are solved quite well. We found that an adaptive method is essential for the efficient solution of these problems.

Finally, in Chapter 8, we showed the relationship between the index of the eigenvalue and the space and time requirements of our algorithm. We found that our method is practical for use in computing more than just the lowest eigenvalues. We noted, however, that these relationships are highly problem dependent. We also showed that our method can outperform an existing initial method in terms of accuracy of error estimates and speed of computation.

We conclude that the automatic, adaptive Sturm-Liouville solution method developed in this thesis is an essential tool for solving the Sturm-Liouville problem. It makes the finite-difference technique competitive with, or preferable to, the other methods being used.

REFERENCES

1. P. B. Bailey, M. K. Gordon, and L. F. Shampine, Solving Sturm-Liouville Eigenvalue Problems, Sandia Laboratories, 1976.

2. D. O. Banks and G. J. Kurowski, Computation of Eigenvalues of Singular Sturm-Liouville Systems, Math. Comp. 22, 304-310.

3. G. Birkhoff and G.-C. Rota, Ordinary Differential Equations, Ginn, Boston, 1962.

4. __________, C. de Boor, B. Swartz, and B. Wendroff, Rayleigh-Ritz Approximation by Piecewise Cubic Polynomials, SIAM J. Numer. Anal. 3 (1966), 188-203.

5. __________ and G. Fix, Accurate Eigenvalue Computations for Elliptic Problems, SIAM-AMS Proceedings Vol. II, Providence, RI, 1970.

6. W. E. Boyce and R. C. DiPrima, Elementary Differential Equations and Boundary Value Problems, John Wiley and Sons, New York, 1969.

7. R. R. Brown, Numerical Solution of Boundary Value Problems Using Nonuniform Grids, J. SIAM 10 (1962), 475-495.

8. J. Canosa and R. G. Oliveira, A New Method for the Solution of the Schroedinger Equation, J. Comp. Phys. 5, 188-207.

9. E. A. Coddington and N. Levinson, Theory of Ordinary Differential Equations, McGraw-Hill, New York, 1955.

10. S. D. Conte and C. de Boor, Elementary Numerical Analysis: An Algorithmic Approach, McGraw-Hill, New York, 1972.

11. R. Courant, Variational Methods for the Solution of Problems of Equlibrium and Vibrations, Bull. Amer. Math. Soc. 49 (1943), 1-23.

12. P. J. Davis and P. Rabinowitz, Numerical Integration, Blaisdell, Waltham, Mass., 1967.

13. C. de Boor and B. Swartz, Collocation at Gaussian Points, SIAM J. Numer. Anal. 10, 582-606.

14. V. E. Denny and R. B. Landis, A New Method for Solving Two-Point Boundary Value Problems Using Optimal Node Distribution, J. Comp. Phys. 9 (1972), 120-137.

15. J. S. Dranoff, An Eigenvalue Problem Arising in Mass and Heat Transfer Studies, Math. Comp. 15, 403-409.

16. C. F. Fischer, The Deferred Difference Correction for the Numerov Method, Com. Phys, Comm. 2 (1971).

17. ____________ and R. A. Usmani, Properties of Some Tridiagonal Matrices and Their Application to Boundary Value Problems, SIAM J. Numer. Anal. 6 (1969), 127-142.

18. L. Fox, The Numerical Solution of Two-Point Boundary Problems in Ordinary Differential Equations, Oxford Univ. Press, 1957.

19. J. Gary and R. Helgason, A Matrix Method for Ordinary Differential Eigenvalue Problems, J. Comp. Phys. 5 (1970), 169-187.

20. C. W. Gear, Numerical Initial Value Problems in Ordinary Differential Equations, Prentice-Hall, Englewood Cliffs, New Jersey, 1971.

21. B. A. Hargrave, Numerical Approximation of Eigenvalues of Sturm-Liouville Systems, J. Comp. Phys. 20 (1976), 381-396.

22. G. Hellwig, Differential Operators of Mathematical Physics, Addison-Wesley, Reading Mass., 1964.

23. P. Henrici, Discrete Variable Methods in Ordinary Differential Equations, John Wiley and Sons, New York, 1962.

24. E. Hille, Lectures on Ordinary Differential Equations, Addison-Wesley, Reading, Mass., 1969.

25. H. B. Keller, Numerical Methods for Two-Point Boundary Value Problems, Blaisdell, Waltham, Mass., 1968.

26. M. Lentini and V. Pereyra, A Variable Order Finite Difference Method for Nonlinear Multipoint Boundary Value Problems, Math. Comp. 28 (1974), 981-1003.

27. __________, An Adaptive Finite Difference Solver for Nonlinear Two Point Boundary Value Problems with Mild Boundary Layers, STAN-CS-75-530, Stanford Univ., 1975.

28. M. A. Malcolm and R. B. Simpson, Local versus Global Strategies for Adaptive Quadrature, ACM Trans. Math. Software 1, June 1975, 129-146.

29. W. H. Mills, Jr., The Resolvent Stability Condition for Spectral Convergence with Application to the Finite Element Approximation of Non-Compact Operators, to appear Math. Comp.

30. ______________, Optimal Error Estimates for the Finite Element Spectral Approximation of Non-Compact Operators, to appear in SIAM J. Numer. Anal.

31. A. Ostrowski, On the Convergence of the Rayleigh Quotient for the Computation of Characteristic Roots and Vectors I, II, III, IV, V, VI, Arch. Rat. Mech. Anal. (1958-1959), Vol. 1, 233-241; Vol. 2, 243-248; Vol. 3, 325-340, 341-347, 472-481; Vol. 4, 153-165.

32. V. Pereyra, High Order Finite Difference Solution of Differential Equations, STAN-CS-73-348, Stanford Univ., 1973.

33. __________ and E. G. Sewell, Mesh Selection for Discrete Solution of Boundary Problems in Ordinary Differential Equations, Numer. Math. 23 (1975), 261-268.

34. G. B. Price, Bounds for Determinants with Dominant Principal Diagonal, Proc. AMS 2 (1951), 497-502.

35. S. Pruess, Estimating the Eigenvalues of Sturm-Liouville Problems by Approximating the Differential Equation, SIAM J. Numer. Anal. 10, 55-68.

36. W. Rudin, Principles of Mathematical Analysis, McGraw-Hill, New York, 1964.

37. R. D. Russell and L. F. Shampine, A Collocation Method for Boundary Value Problems, Numer. Math. 19, 1-28.

38. ____________ and J. M. Varah, A Comparison of Global Methods for Linear 2-point Boundary Value Problems, Math. Comp. 29, 1007-1019.

39. G. W. Stewart, Introduction to Matrix Computations, Academic Press, New York, 1973.

40. E. C. Titchmarsh, Eigenfunction Expansions Associated with Second-Order Differential Equations, I, Oxford, 1946.

41. R. S. Varga, Matrix Iterative Analysis, Prentice-Hall, Englewood Cliffs, New Jersey, 1962.

42. J. H. Wilkinson, The Algebraic Eigenvalue Problem, Clarendon Press, Oxford, 1965.

VITA

David Alan Nelson was born October 25, 1947 in York, Pennsylvania. He received a B. S. in Mathematics and Computer Science from the United States Air Force Academy in 1969 and a Masters of Applied Mathematics from North Carolina State University in 1970. He attended The Pennsylvania State University from September 1975 to February 1978 pursuing the degree of PhD in Computer Science.

He is currently a captain in the United States Air Force and an Assistant Professor of Mathematics at the United States Air Force Academy.